TAMPEREEN TEKNILLINEN YLIOPISTO

Noora Vainio

Testauspolitiikan kehittäminen alihankintaprojekteihin

Diplomityö

Tarkastaja: Prof. Tommi Mikkonen
Tarkastaja ja aihe hyväksytty
Tieto- ja sähkötekniikan
tiedekunnan dekaanin toimesta
26. huhtikuuta 2017

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietotekniikan diplomi-insinöörin tutkinto-ohjelma
Noora Vainio: Testauspolitiikan kehittäminen alihankintaprojekteihin
Diplomityö, 45 sivua, 4 liitesivua
Kesäkuu 2017
Pääaine: Ohjelmistotuotanto
Tarkastajat: Prof. Tommi Mikkonen
Avainsanat: Alihankintaprojektien testaus, testaus, testauspolitiikka, testausstrategia

Alihankintaprojekteja tekevissä yrityksissä projektien diversiteetti on suuri. Projektien testausvaatimukset ja -käytännöt vaihtelevat tämän vuoksi suuresti. Kuitenkin asiakkaat ovat erityisesti myyntivaiheessa kiinnostuneita yrityksen testauskäytännöistä. Lisäksi projektien laadun pitäisi pysyä tasaisena. Testauspolitiikka eli organisaation laajuiset testauskäytännöt voivat auttaa näihin ongelmiin.

Tässä työssä kehitettiin yritykselle testauspolitiikka pohjautuen olemassa oleviin testauskäytäntöihin. Tavoitteena oli luoda käytännönläheinen ja yleispätevä testauspolitiikka, jota voidaan hyödyntää erilaisissa projekteissa. Testauspolitiikka muodostettiin yrityksen työntekijöiden haastatteluiden pohjalta.

Työn tavoitteena oli myös arvioida testauspolitiikan sopivuutta testausstrategian muodostamiseen. Projektin testausstrategialla tarkoitetaan yhden projektin lähestymistapaa testaukseen, jossa kuvataan mm. testauskäytännöt, testiympäristö, testiaineistot ja testaustyökalut.

Testauspolitiikan luomisen jälkeen sitä sovellettiin esimerkkiprojektin testausstrategian luomiseen. Projektin testausta seurattiin ja projektihenkilökuntaa haastateltiin projektin lopussa testauskäytäntöihin sekä testauspolitiikkaan liittyen. Projekti pohjautui vahvasti aiemmin toteutettuun järjestelmään, minkä vuoksi kehitys- ja testauspanos projektissa jäi hyvin pieneksi. Näin ollen esimerkkiprojekti osoittautui liian pieneksi.

Testauspolitiikan luominen haastatteluiden pohjalta onnistui, ja esimerkkiprojektin henkilökunta koki testauspolitiikan sopivan yleispäteväksi alihankintaprojekteihin. Esimerkkiprojektin seurannasta ei voitu kuitenkaan muodostaa luotettavaa arviota testauspolitiikan hyödystä testausstrategian luomiseen, koska aineistoa ei saatu kerättyä riittävästi. Jotta testauspolitiikan hyödyistä kohdeyrityksessä saisi muodostettua luotettavamman arvion, tulisi esimerkkiprojekteja olla enemmän ja niiden oltava laajempia. Työn lopussa ehdotetaan jatkomahdollisuuksia testauspolitiikan hyödyntämisen arvioimiseen luotettavammin.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
Noora Vainio: Testing policy development for subcontracting projects
Master of Science Thesis, 45 pages, 4 appendix pages
June 2017
Major: Software Engineering
Examiners: Prof. Tommi Mikkonen
Keywords: Software testing in subcontracting projects, software testing, testing policy, testing strategy

Companies undertaking subcontracting projects have great diversity in projects. Thus, testing requirements and practices differ a lot. However, especially in the sales phase the customer is interested in company's testing practices. Also, the quality of the projects should remain stable. Testing policy ergo company wide testing practices could help with these problems.

In this thesis, a testing policy was developed for a company based on its existing testing practices. The goal was to define a practical and general testing policy which can be utilized in different kinds of projects. Testing policy was developed based on interviews of the company's employees.

The goal was also to evaluate the testing policy in testing strategy development. Testing strategy is a testing approach for one project which describes testing practices, test environment, test data and test tools, among other things.

After developing the testing policy, it was utilized for defining a testing strategy for a case project. Testing in the case project was monitored, and in the end of the project the project team was interviewed about testing practices and testing policy. Project was heavily based on a pre-existing project, and therefore development and testing contributions were small. Hence the case project turned out to be too inadequate.

Defining a testing policy based on interviews was successful. The project team in the case project felt that the testing policy was general enough for subcontracting projects. Monitoring the case project didn't produce enough data to formulate a reliable evaluation of benefits of testing policy in defining testing strategy. To create a better evaluation of benefits of the testing policy in the company, there should be more and bigger case projects. In the end of this thesis some future possibilities are presented to evaluate the utilization of the testing policy.

ALKUSANAT

Tämä diplomityö on tehty Solita Oy:ssä työskennellessäni. Suuret kiitokset Solitalle tämän työn mahdollistamisesta ja etenkin Pekka Marjamäelle aiheen ideoinnista. Kiitos myös kaikille haastatteluihin ja esimerkkiprojektiin osallistuneille Solitalaisille.

Erityinen kiitos myös ohjaajalleni Samuli Lahnamäelle ja työn tarkastajalle Tommi Mikkoselle. Ilman teidän palautettanne ja ohjausta työn valmistuminen ei olisi ollut mahdollista. Kiitokset työn oikoluvusta Karoliina Jäspille ja Noora Pirttilahdelle.

Haluan kiittää myös ystäviäni, joiden ansiosta koko opiskeluaikani on ollut uskomattoman hienoa. Erityisesti kiitos Jaakolle tuesta ja kannustuksesta diplomityöprojektini ajan.

Tampere, 14.5.2017

Noora Vainio

SISÄLLYS

1. JOHDANTO

Alihankintaprojekteja tekevissä ohjelmistoalan yrityksissä yhtenäisten testauskäytäntöjen määrittäminen on vaikeaa projektien diversiteetin vuoksi. Ilman yhteisiä testauskäytäntöjä projektit joutuvat usein suunnittelemaan projektin lähestymistavan testaukseen itsenäisesti. Näin ollen testauksen taso saattaa vaihdella projektien välillä merkittävästi. Yritykselle olisi kuitenkin tärkeää pitää järjestelmiensä laadunvarmistus ja laatu samalla tasolla koko organisaation laajuisesti. Lisäksi asiakkaat ovat etenkin myyntivaiheessa kiinnostuneita yrityksen testauskäytännöistä. Tätä varten joudutaan kuvaamaan yrityksen testausta aina uudestaan jokaista tarjousta varten. Nämä ongelmat voitaisiin ratkaista testauspolitiikan eli yrityksen laajuisten ohjenuorien avulla.

Tässä työssä ongelmaa lähdetään ratkaisemaan alhaalta päin eli testausprosessissa mukana olevien henkilöiden tarpeisiin ja näkemyksiin pohjautuen. Kohdeyritykselle pyritään määrittämään yhtenäiset testauskäytännöt, joita on mahdollista soveltaa useissa erilaisissa projekteissa ja joihin yrityksen työntekijät ovat valmiita sitoutumaan. Kohdeyrityksessä testausta tehdään paljon kontekstipohjaisesti. Kontekstipohjainen testaus on Bachin, Marickin, Pettichordin ja Kanerin kehittämä lähestymistapa testaukseen [1]. Kontekstipohjaisessa testauksessa testauksen tavoitteet, tekniikat ja lopputuotteet valitaan tietyn tilanteen yksityiskohtien sekä testauksen tilanneiden sidosryhmien tahtojen perusteella [1].

Testauspolitiikan muodostamisen jälkeen työssä selvitetään sen sopivuutta yksittäisen projektin testaussuunnitelman muodostamiseen. Testauspolitiikan pohjalta luodaan esimerkkiprojektille sopiva testausstrategia. Testausstrategian toimivuutta ja testauspolitiikan hyödyllisyyttä seurataan projektin aikana. Tämän seurannan perusteella muodostetaan arvio testauspolitiikan sopivuudesta kohdeyrityksen testauksen suunnitteluun.

Luvussa 2 käsitellään testauspolitiikan ja -strategian määritelmät ja esitellään me-

netelmiä näiden luomiseen. Lisäksi käsitellään aiheeseen liittyviä aikaisempia tutkimuksia. Luvussa 3 esitellään menetelmä testauspolitiikan kehittämiseen ja tällä menetelmällä kohdeyritykselle kehitetty testauspolitiikka. Luku 4 käsittelee testausstrategian kehittämistä kohdeyritykselle luodun testauspolitiikan perusteella esimerkkiprojektissa. Luvussa 5 arvioidaan testauspolitiikkaa ja sen hyödyntämistä testausstrategian luomisessa kokonaisuudessaan ja esitetään mahdolliset jatkokehitysideat. Lopuksi luvussa 6 on yhteenveto työstä ja sen tuloksista.

2. ARVOT, TESTAUSPOLITIIKKA JA -STRATEGIA

Tässä luvussa esitellään testauspolitiikan ja -strategian määritelmät ja näiden määrittämiseen liittyvää teoriaa sekä aiempia tutkimuksia liittyen testauspolitiikkaan. Kohdassa 2.1 määritellään testauspolitiikka ja kohdassa 2.2 käsitellään organisaation arvoja sekä testauksen arvoihin liittyviä aiempia tutkimuksia. Kohdassa 2.3 kuvataan yksi menetelmä testauspolitiikan luomiseen. Testausstrategia määritellään kohdassa 2.4, ja kohta 2.5 esittelee menetelmän testausstrategian muodostamisen.

2.1. Testauspolitiikka

Testauspolitiikka on määritetty muun muassa IEEE:n (Institute of Electrical and Electronics Engineers) standardissa 29119-1 [2] ja ISTQB:n (International Software Testing Qualifications Board) sanastossa [3]. IEEE standardi 29119-1 määrittää organisaation testauspolitiikan liikkeenjohdollisena dokumenttina, joka kuvaa testauksen tarkoituksen, tavoitteen ja yleisen laajuudeen organisaation sisällä [2]. IEEE:n mukaan dokumentti myös kuvaa sitä, miksi testataan ja mitä testauksella pyritään saavuttamaan. ISTQB puolestaan määrittelee testauspolitiikan korkean tason dokumenttina, joka kuvaa organisaation sisäiset testauksen periaatteet, lähestymistavan ja päätavoitteet [3].

Tässä työssä testauspolitiikkaa käsitellään organisaation laajuisena testauksen ohjenuorana, joka sisältää yleisiä ohjeita testaukseen. Lisäksi testauspolitiikka voi kuvata organisaation testaukseen liittyviä arvoja.

2.2. Organisaation arvot

Tom Kenny määrittää arvot yksilön, ryhmän, organisaation tai yhteisön standardeina tai periaatteina [4]. ISTQB:n määritelmän mukaan testauspolitiikka kuvaa

testauksen periaatteet organisaation tasolla, eli testauspolitiikka voidaan käsittää myös testauksen arvoina.

Kennyn mukaan arvot heijastavat yksilön tai yhteisön arviota siitä, mikä on tärkeää tai arvostettavaa elämässä. Näistä arvioista saadaan mittari, jota vasten voidaan arvioida henkilön, organisaation tai yhteisön käytöstä. Samoin testauspolitiikka heijastaa sitä, mikä on testauksessa tärkeää tai arvostettavaa. Testauspolitiikka vasten voidaan siis arvioida organisaation testauskäyttäytymistä.

Arvot ovat lähtökohtaisesti abstrakteja, epämääräisiä ja subjektiivisia [4]. Tästä huolimatta Kennyn mukaan yrityksen johdon sekä yksilön ja organisaation käytöksen tulisi perustua arvoihin. Seuraavassa luettelossa listataan ominaisuudet, jotka Kennyn mukaan ovat niillä arvoilla, joiden hyödyntämispotentiaali on suurin:

- eksplisiittisuus,
- spesifisyys ja konkreettisuus,
- yksitulkintaisuus,
- arvot ovat jaettuja ja niihin ollaan sitouduttu,
- käytännöllisyys ja operatiivisuus,
- objektiivisuus ja standardoituminen,
- arvojen on oltava suunnitteltuja, toteutettuja, valvottuja ja arvioituja.

Organisaation testaukseen liittyviä arvoja on selvitetty aikasemmissa tutkimuksissa. Mäntylä et al tutkivat testausorganisaatioita ja selvittivät samalla esimerkkiyritysten testauksen arvoja. Tutkimuksessa käytettiin menetelmänä puolistrukturoituja haastatteluita. Haastatteluiden avulla saatiin selville muun muassa se, kuinka paljon työntekijät arvostavat testaajan teknistä osaamista ja kokemusta, minkälaiset ominaisuudet ovat tärkeimpiä ohjelmistojen laadussa sekä kuka ohjelmistoja oikeasti testaa. Tuloksissa nousi esiin tärkeinä huomiona testaajan toimialuetietämys ja asiakkaan suorittama testaus. Toimialuetietämyksellä tarkoitetaan tietämystä testattavan ohjelmiston käyttämisestä ja kontekstista. [5]

Martin et al kuvasivat kuinka organisaation prioriteetit vaikuttavat testauksen arvoihin ja kuinka testausta käytännössä suoritetaan. He painottivat tuloksissaan testauksen sosioteknistä luonnetta. Heidän mukaansa testaukseen liittyvissä aktiviteeteissä on käytännössä huomioitava asiakassuhdedynamiikat, kuinka hyödynnetään rajallinen panos mahdollisimman tehokkaasti ja julkaisuiden ajoittaminen. Martin et al kartoittivat myös ohjelmistojen testauksesta tehtyjä empiirisiä tutkimuksia ja havaitsivat niiden määrän vähäiseksi. [6]

Rooksby et al tekemässä etnograafisessa tutkimuksessa kuvataan testauksen sosiaalisia ja organisationaalisia ulottuvuuksia. Heidän mukaansa on liian vähän tutkimusta siitä, kuinka testausta tehdään oikeassa maailmassa. Tulokset osoittavat monien testauksen ongelmien aiheutuvan organisaatiosta ja käytännön asioista. Rooksby et al uskovat organisaation rakenteen ja prioriteettien vaikuttavan käytännön testaukseen. [7]

Andersson ja Runeson tutkivat testausprosesseja 11 ruotsalaisessa yrityksessä kvalitatiivisin metodein. Heidän mukaansa arvot verifointia ja validointia kohtaan ovat kunnianhimoisia, mutta useimmiten valinta on ajan ja laadun välillä. Tulosten mukaan testauskäytännöt ovat riippuvaisia yrityksen koosta ja organisaation rakenteesta sekä asiakkaista ja valmistettavista tuotteista. Anderssonin ja Runesonin mukaan yritykset kokevat tarvitsevansa kokenutta testaushenkilökuntaa. [8]

Beer ja Ramler tutkivat kokemuksen merkitystä testauksessa. Heidän tavoitteenaan oli selvittää, mihin kokemusta hyödynnetään testauksessa, mikä arvo kokemuksella on testauksessa, mitä kokemukseen pohjautuvia testausmenetelmiä käytetään, kuinka testauskokemus on kerrytetty sekä miten kokemusta hallitaan ja kehitetään yrityksessä. Beer ja Ramler suorittivat kohdeyrityksissä haastatteluita sekä analysoivat projektien materiaaleja. Heidän löydöksensä osoittavat toimialuetietämyksen olevan tärkeää testaustietämyksen lisäksi. [9]

Itkonen ja Rautiainen tutkivat kolmea yritystä, jotka hyödynsivät tutkivaa testausta. Tutkivassa testauksessa ei ole ennalta määrättyjä testitapauksia, vaan testaaja samanaikaisesti oppii, suunnittelee ja testaa [10]. Itkosen ja Rautiaisen tavoitteena oli selvittää, millaista kirjallisuutta tutkivasta testauksesta on sekä miten ja miksi yritykset hyödyntävät tutkivaa testausta. Heidän tuloksissaan korostui testaa-

jan toimialuetietämys ja loppukäyttäjän näkökulmasta testaaminen. Lisäksi tärkeänä huomiona nousi esiin, kuinka tutkiva testaus mahdollistaa testaajan kokemuksen ja luovuuden hyödyntämisen sekä nopean palautteen testituloksista kehittäjille. [11]

Taipale et al tutkivat testauskäytäntöjä organisaation ja tiedon hallinnan näkökulmasta. He tutkivat useita organisaatioyksiköitä, jotka kehittävät ja testaavat ohjelmistoja automaation ja tietoliikennetekniikan aloilla. Tutkimusten tulokset esittävät organisaation liiketoiminnallisen suuntauksen vaikuttavan testausorganisaatioon, testaustiedonhallintaan ja tiedon välitykseen organisaatioyksiköiden välillä. Taipale et al selvittivät myös testauksen eroja tuotetalojen ja palveluliiketoimintaa harjoittavien yritysten välillä. Heidän mukaansa palveluliiketoimintaan keskittyneet yritykset mukautuvat useammin asiakkaan testausprosesseihin ja tarvitsevat enemmän toimialuetietämystä. Lisäksi palveluliiketoimintaa harjoittavien organisaatioiden on vaikeampaa ulkoistaa testausta, ja heillä on enemmän ongelmia sisäisessä tiedon hallinnassa sekä välittämisessä. [12]

2.3. Testauspolitiikan luominen

Testauspolitiikalle ei ole yhtä ainoaa formaattia, joten sen luomiselle ei ole tiettyä prosessia. Pinkster kuvaa testauspolitiikan keinona välittää yrityksen strategia ohjenuorina, jotka määrittävät organisaation sisäisen testauksen [13]. Hänen mukaansa testauspolitiikan tulee sisältää testauksen tavoitteet ja suuntaviivat sekä viitekehys, jonka avulla tuotteen laatua ja testausprosessia voidaan mitata.

Pinksterin mukaan testauspolitiikan määrittäminen koostuu kymmenestä vaiheesta [13]. Ensimmäisenä määritetään testauspolitiikan avainsanat eli yrityksen arvot ja yrityksen luonnetta kuvaavat sanat. Tämän jälkeen avataan tarkemmin, mitä nämä avainsanat tarkoittavat. Kolmannessa vaiheessa selvitetään, mikä olisi pahinta, mitä voi tapahtua. Vaiheessa neljä määritetään, minkä on pakko toimia tuotteessa. Viidenneksi selvitetään testauspoliitikan vaikutus budjettiin. Seuraavaksi kartoitetaan organisaation tuottamat palvelut sekä tuotteet, ja määritetään niille tärkeysjärjestys. Seitsemännessä vaiheessa annetaan suuntaviivat testauksen määrälle. Tämän jälkeen selvitetään yleisellä tasolla testauksen työkaluja ja lähestymistapoja. Lopuksi arvioidaan lopputulosta ja raportoidaan se.

Pinksterin mukaan testauspolitiikkaa voidaan lähteä luomaan joko alhaalta ylös tai ylhäältä alas. Alhaalta ylös testauspolitiikan tarve ja toiveet nousevat henkilöstöltä, jotka ovat osallisena testausprosessissa. Alhaalta ylös lähestymisen hyvät puolet ovat Pinksterin mukaan työntekijöiden sitoutuminen, tärkeimpien asioiden selkeä identifiointi ja konkreettisten tulosten nopea saavuttaminen. Huonoiksi puoliksi Pinkster kokee johdon tuen puutteen, pitemmän tähtäimen jatkuvuuden epävarmuuden, menetelmän tehottomuuden, epäoleellisen työn riskin ja mahdolliset konfliktit muiden organisaation laajuisten politiikkojen kanssa. [13]

Ylhäältä alas testauspolitiikan lähtökohtana on ylemmän johdon määrittämä strategia ja visiot. Pinkster esittää ylhäältä alas lähestymistavan hyviksi puoliksi yhdenmukaisuuden organisaation muiden politiikkojen kanssa, ylimmän tason tuen sekä johdonmukaisuuden yrityksen strategian, mission ja visioiden kanssa. Huonoina puolina hän näkee haasteet uusien käytäntöjen sovittamisessa nykyisiin, hitaan toteutumisen, epäsopivuuden nykytilaan ja tuen puutteen liiketoimintatason organisaatiolta. [13]

Pinkster suosittelee yhdistämään alhaalta ylös ja ylhäältä alas -menetelmät, jolloin käytännön toimenpiteet saadaan sovitettua parhaiten yrityksen strategiaan ja tavoitteisiin. Pinkster painottaa testauspolitiikan luomisen olevan evolutionäärinen prosessi, jossa ensimmäisellä yrityksellä ei saavuteta täydellistä versiota. [13]

2.4. Testausstrategia

Samoin kuin testauspolitiikalle, testausstrategialle löytyy lukuisia määritelmiä. IEEE:n standari 29119-1 määrittää organisaation laajuisen testausstrategian dokumenttina, joka ilmaisee yleiset vaatimukset kaikkien projektien testaukselle [2]. Standardissa huomioidaan myös konteksteiltaan merkittävästi erilaiset projektit. Organisaatiolla voikin olla useita erilaisia testausstrategioita kattamaan myös huomattavan erilaiset testaustarpeet.

IEEE:n standardi 29119-1 määrittelee testaussuunnitelman yksityiskohtaisena kuvauksena testauksen tavoitteista sekä keinoista ja aikatauluista, joilla nämä tavoitteet saavutetaan [2]. Standardin mukaan testaussuunnitelman tarkoitus on organisoida yhden testikohteen tai testikohdejoukon testausaktiviteetit.

Tässä työssä testausstrategiaa käsitellään yksittäisen projektin testausstrategiana. IEEE:n standardi ohjelmistojen testaukseen kuvaa testausstrategian testaussuunnitelman osana, joka kuvaa lähestymistavan kyseisen projektin testaukseen [2]. Standardin mukaan testausstrategia kuvaa yleensä seuraavat asiat: testauskäytännöt, toteutettavat testauksen aliprosessit, uudelleen testauksen sekä regressiotestauksen, testien suunnittelutekniikat sekä vastaavat testien läpäisykriteerit, testiaineiston, testiympäristön, testaustyökalut ja odotukset testauksen lopputuotoksille.

ISTQB:n testaussanasto puolestaan määrittelee testausstrategian organisaation laajuisena korkean tason kuvauksena testauksen tasoista ja testauksesta, jota kullakin tasolla suoritetaan [3]. ISTQB:n sanasto käyttää termiä testauslähestymistapa kuvaamaan organisaation laajuisen testistrategian jalkauttamista yksittäiseen projektiin.

2.5. Testausstrategian luominen

Caner et al teoksessa "Lessons Learned in Software testing" esitellään käytännön vinkkejä testausstrategian luomiseen ja hyödyntämiseen [10]. He määrittävät testausstrategian yhteytenä käytännön testauksen ja projektin tavoitteen välillä. Kirjassa esitetään kolme peruskysymystä, joita tulee kysyä jatkuvasti testausstrategiaa luodessa. Kysymykset ovat seuraavat:

- Miksi vaivautua?: Testaaminen on kallista, joten ei kannata testata turhia asioita.
- Keitä kiinnostaa?: Kannattaa testata vain asioita, joiden toimivuudella on väliä jollekin sidosryhmälle.
- Kuinka paljon?: Kuinka paljon mitäkin aiotaan oikeasti testata.

Caner et al mukaan hyvä testausstrategia selittää ja oikeuttaa projektissa suoritettavan testauksen [10]. Ei ole oleellista, millä tavoin testausstrategia on dokumentoitu, vaan esitysmuoto on sovitettava projektille sopivaksi. Muutenkin heidän mukaansa on oleellista huomioida projektin konteksti. Kontekstista on huomioitava ainakin seuraavat seikat:

- Kehitys: kuinka testattava tuote saadaan käyttöön? Missä vaiheessa kehitystä tuote on?
- Vaatimukset: minkälaisia vaatimuksia valmiilla tuotteelle on? Mitä riskejä projektilla on? Kenen mielipiteellä on merkitystä?
- Testaustiimi: ketkä testaavat? Minkälaisia taitoja testaajilla on?
- Testausympäristö: mitä työkaluja ja materiaaleja on käytettävissä? Mihin virheet ja havainnot kirjataan?
- Tavoite: mikä on testauksen tavoite?

Hyvä testausstrategia selventää testausprosessiin liittyviä valintoja. Caner et al mukaan strategiaan liittyvistä valinnoista olisi hyvä kuvata, mitä aiotaan testata, mitä testaustekniikoita aiotaan käyttää ja kuinka virheet tunnistetaan. Heidän mukaansa hyvä testausstrategia on enemmän kuin suoritettavat testit, eli se kuvaa testaustekniikat ja -lähestymistavat. Caner et al listaavat hyvän testausstrategian ominaisuuksiin myös tuotespesifisyyden, riskeihin keskittymisen, monipuolisuuden ja käytännöllisyyden. Tuotespesifisyydellä tarkoitetaan sitä, että testausstrategia on sovitettu kehitettävään tuotteeseen ja käytettyihin teknologioihin. Riskeihin keskittymällä testataan tärkeimpiä asioita, jolloin testauksesta saadaan tehokasta ja kannattavaa. Riskipohjaisuudessa on huomioitava myös testausstrategian päivittäminen projektin edetessä, sillä projektin riskit tarkentuvat ja muuttuvat samalla kun saadaan lisää tietoa projektista. Monipuolisuus näkyy testaussuunnitelmassa useina erilaisina tekniikoina. Caner et al uskovat, että on kannattavampaa testata useammasta näkökulmasta osittain kuin yhdestä näkökulmasta täydellisesti. Testausstrategian käytännöllisyys tarkoittaa, että se on realistisesti toteutettavissa. [10]

Testausstrategiassa on huomioitava erilainen testaus projektin eri vaiheissa. Projektin alkuvaiheessa on testattava sympaattisemmin, keskivaiheilla aggressiivisemmin sekä laajemmin ja lopussa mahdollisimman tarkasti. Projektin jokaisessa vaiheessa on syytä miettiä, mitä voi testata ja miten voi testata. Caner et al suosittelee testaustasojen määrittelemistä. Heidän esimerkkitasonsa on esitelty taulukossa 2.1. Nämä tasot ovat: savutestaus, kyvykkyystestaus, toiminnallisuustestaus

Taso	Miten testatataan
Taso 0: Savutestaus	Varmistetaan yksinkertaisilla testeillä, onko ohjelma valmis testattavaksi.
Taso 1: Kyvykkyystestaus	Varmistetaan, että yksittäinen ominaisuus selviää tehtävästään perustilanteissa.
Taso 2: Toiminallisuustestaus	Varmistetaan toiminallisuuden oikeus ja luotettavuus. Testataan virhetilanteita ja laajalla datajoukolla, mutta vältetään monimutkaisia tilanteita ja toimintojen interaktioita.
Taso 3: Monimutkainen testaus	Testataan interaktioita ja monimutkaisia tilanteita jotka muodostuvat toimintoryhmistä. Laajempaa luotettavuuden testausta, joka on mahdollista vasta projektin ikääntyessä.

Taulukko 2.1: Caner et al esimerkkitasot testaukseen

ja monimutkainen testaus. Savutestauksella tarkoitetaan yksinkertaisia testejä, joilla varmistetaan onko ohjelma valmis testaukseen. Kyvykkyystestauksessa varmistetaan yksittäisen ominaisuuden toiminta perustilanteessa. Toiminnallisuustestauksella varmistetaan toiminnallisuuden oikeus ja luotettavuus. Viimeisellä tasolla eli monimutkaisessa testauksessa testataan interaktioita ja monimutkaisia tilanteita, jotka koostuvat toimintoryhmistä. [10]

3. SOLITAN TESTAUSPOLITIIKKA

Tässä luvussa kuvataan, kuinka kohdeyritykselle määritettiin testauspolitiikka. Testauspolitiikka muodostettiin yrityksen olemassa olevien testauskäytäntöjen pohjalta. Testauspolitiikasta pyrittiin saamaan mahdollisimman käytännöllinen ja yleispätevä. Lisäksi testauspolitiikan toivottiin olevan projekteissa oikeasti sovellettava, joten päädyttiin alhaalta ylöspäin lähestymistapaan. Alhaalta ylöspäin -lähestymistapa sopii myös paremmin yritykselle, sillä yrityksen projekteilla on pääosin autonomia päättää omista toimintatavoistaan. Kohdeyrityksestä on lisätietoja kohdassa 3.1.

Testauspolitiikan muodostamiseksi haastateltiin viittä yrityksen työntekijää, jotka ovat erilaisissa rooleissa. Tämän haastatteluista kerätyn laadullisen aineiston perusteella muodostettiin hypoteesit yrityksen nykyisistä testauskäytännöistä. Haastattelut esitellään kohdassa 3.2 ja näistä saadun aineiston analyysi kuvataan kohdassa 3.3. Haastatteluaineiston pohjalta muodostettu testauspolitiikka esitellään kokonaisuudessaan kohdassa 3.4.

3.1. Kohdeyritys

Solita Oy (myöhemmin yritys) on vuonna 2006 perustettu digitaalisen alan palveluyritys. Yrityksen toimialoja ovat verkkopalvelut ja -kaupat, digitaalisten palveluiden kehitys, sähköinen asiointi, ennakoiva analytiikka, tiedolla johtaminen, informaation hallinta ja big data, digitaalinen strategia ja palvelumuotoilu, integraatioratkaisut, Internet of Things ja teollinen internet sekä Solita Cloud Services [14]. Tässä tutkimuksessa toimialoista keskityttiin lähinnä verkkopalveluiden kehittämiseen. Tämän diplomityön tekemisen aikana yrityksessä oli noin 500 työntekijää. Yrityksen arvoja ovat rohkeus, rentous, intohimo ja välittäminen. Yritys kertoo toimivansa asiakkaidensa matkaoppaina digitalisoituvassa maailmassa.

3.2. Haastattelut

Tutkimuksen tavoitteena on kerätä laadullista dataa, johon sopiva menetelmä on haastattelut. Haastettelutyyppinä käytettiin puolistrukturoitua haastattelua. Puolistrukturoidussa haastattelussa kysymykset ovat ennalta määrättyjä, mutta kysymyksiä voidaan kysyä myös ennaltamäärätyn kysymyssetin ulkopuolelta. Tämä mahdollistaa tarkentavat lisäkysymykset ja keskittymisen haastateltavalle tärkeimpiin asioihin. [15]

Yrityksen laajan palvelutarjoaman vuoksi tutkimuksen otosta rajattiin vertailtavuuden saavuttamiseksi. Haastatteluissa keskityttiin pieniin web-teknologioilla toteutettaviin projekteihin. Pienellä projektilla tarkoitetaan tässä projektia, jonka kesto on alle vuosi ja projektin henkilöstömäärä on kokonaisuudessaan 3–7 henkeä.

Haastatteluissa pyrittiin kattamaan projektin koko elinkaari pelkän toteutusvaiheen sijasta. Vaikka testaus käytännössä suunnitellaan ja toteutetaan projektin toteutusvaiheessa, myös myynti- ja sopimusneuvotteluvaiheissa sekä ylläpidossa on huomioitava testaaminen.

Projektin myyntivaiheessa muodostetaan tarjouksia ja kirjoitetaan ratkaisukuvauksia asiakkaan toivomille järjestelmille. Myynti- ja sopimusneuvotteluissa tehdään usein myös alustavat työmääräarviot. Tarjousten ja ratkaisukuvausten perusteella asiakas valitsee järjestelmälle toteuttajan.

Myyntivaihetta seuraa projektin toteutusvaihe, jossa tapahtuu järjestelmän kehitys. Toteutusvaiheessa projektiin osallistuvat yleensä kehittäjät ja projektipäällikkö sekä mahdolliset testaajat. Nämä ovat henkilöitä, joiden vastuulla testaaminen käytännössä on. Kohdeyrityksessä projektit ovat useimmiten autonomisia, ja ne voivat itse päättää sopivista käytännöistä projektin sopimuksen puitteissa.

Ylläpitovaiheella tarkoitetaan sitä vaihetta, jossa järjestelmä on valmis ja kehitystiimi on siirtynyt pääasiallisesti muihin tehtäviin. Useimmat yrityksen projektit jäävät kuitenkin yrityksen ylläpidettäviksi, eli käytön aikana havaittavat virhetilanteet on korjattava. Lisäksi ylläpidossa oleviin järjestelmiin saatetaan tehdä pienkehitystä havaittujen tarpeiden mukaan. Ylläpidosta kohdeyrityksessä vastaa tyypillisesti palvelupäällikkö.

Haastateltavat valittiin kartoittamalla tutkimuksen kohderyhmään sopivia hen-

kilöitä. Ennen haastatteluja jokaisen haastateltavan kanssa keskusteltiin tutkimuksesta sekä sen tavoitteista ja näkökulmasta. Tämän perusteella saatiin valittua haastateltavat niin, että projektien koko elinkaari saatiin katettua ja haasteteltavilla oli kokemusta rajaukseen osuvista projekteista. Haastateltavien taustat ovat esitelty taulukossa 3.1.

Haastattelukysymykset jaoteltiin karkeasti seuraaviin teemoihin: testauspolitiikka, prosessit ja työkalut, roolit sekä haastateltavan tausta. Testauspolitiikkakysymykset käsittelivät testausta ja sen merkitystä laajemmassa mittakaavassa. Prosessit ja työkalut -osiossa selvitettiin yrityksen nykyisiä käytäntöjä projektitasolla. Rooleihin liittyvillä kysymyksillä selvitettiin testaukseen liittyviä rooleja ja testaajissa arvostettuja ominaisuuksia. Lopuksi selvitettiin haastateltavan taustaa. Haastattelukysymykset ovat liittessä 1.

3.3. Testauspolitiikan muodostaminen

Haastettelut nauhoitettiin ja tämän jälkeen litteroitiin. Litteroituja haastatteluaineistoja analysoitiin grounded theory -menetelmällä [16]. Menetelmä sopii laadullisen tiedon analysointiin ja uusien teorioiden löytämiseen.

Grounded theory -menetelmällä kehitettiin haastatteluaineistoista löytyvien havaintojen koodauksen ja kategorisoinnin avulla teorioita. Menetelmässä aineistoa analysoidaan systemaattisesti etsimällä siitä koodeja ja yhdistelemällä niitä. Aineiston analyysi aloitetaan heti, kun sitä on saatavilla ja lopetetaan, kun uusia kategorioita ei enää synny. Ensimmäisenä analyysissä suoritetaan avoin koodaus eli kategorioiden ja koodien nimeäminen. Tämän jälkeen aineisto ryhmitellään kategorioiden ympärille eli tehdään selektiivinen koodaus. Lopuksi kategorioista muodostetaan lopulliset teoriat. [16]

Ensimmäisten haastattelujen analyysi aloitettiin heti haastatteluiden jälkeen ja analyysiä jatkettiin ristiin uusien haastattelujen jälkeen. Grounded theory -menetelmässä aineiston keräämistä tulee jatkaa, kunnes uusia koodeja tai kategorioita ei löydy [16]. Kohdeyrityksessä projektit voivat vapaasti valita työtapansa, jonka seurauksena kaikilla työntekijöillä on erilaisia kokemuksia, eli jokaisesta haastattelusta tulisi löytymään jotain uutta. Kuitenkaan viidennestä haastattelusta ei

Nimike	Työvuodet yrityksessä / kokonaisuudessaan	Projektien lkm yrityksessä	Työtehtävät	Aiemmat työtehtävät
Public Sector Analyst	1 vuosi / 10 vuotta /	n. 20	Tarjousten tekeminen kilpailutuksiin	Laajasti IT-alalla mm. testaaja, projektipäällikkö, tiimipäällikkö, määrittelyvastaava
Senior Integration Architect	15,5 vuotta / 17 vuotta	n. 20	Integraatioiden suunnittelua ja määrittelyä, ohjelmistokehitystä -ja testausta, konsultointia, projektinhallintaa sekä organisointia.	Ohjelmistokehitys, tietovarastointi ja raportointi
Software Architect, tiimipäällikkö	3 vuotta / 17 vuotta	4 isompaa, lisäksi vaikuttanut useisiin muihin hiukan	Puolet ajasta esimiestyötä. Muuten ohjelmistojen arkkitehtuurin suunnittelua ja määrittelyä sekä jonkin verran ohjelmointia.	Ohjelmistokehittäjä, testaaja
Service Manager	4,5 vuotta / 4,5 vuotta	n. 10	Projektin ylläpidosta vastaaminen ja sen organisointi	Samassa yrityksessä service deskissä eli ensivasteessa
Project Manager	3 vuotta / 23 vuotta	n. 10	Projektista vastaaminen yhdessä asiakkaan kanssa.	Ohjelmistokehittäjä, testaaja, konsultti

Taulukko 3.1: Haastateltavat

Lopullinen kategoria	Alustavat kategoriat
Testiautomaatio	Testiautomaatio Ympäristöt
Dokumentaatio	Dokumentaatio
Kehittäjälähtöisyys	Kehittäjälähtöisyys Ei erillistä testaajaa Itseohjautuvuus Työkalut
Ylläpidettävyys	Ylläpidettävyys Pareittain testaus ja katselmointi
Osaamisen jakaminen	Osaamisen jakaminen Testaajan taidot ja ominaisuudet
Vastuu laadusta	Testauksen merkitys Vastuu laadusta Vastuu testauksesta
Asiakas	Asiakas
Projektikohtaisuus	Projektien autonomia Testaamiseen vaikuttavat tekijät Erilaiset tavat testata Testauksen suunnittelu

Taulukko 3.2: Selektiivisen koodauksen kategoriat

löytynyt enää uusia isoja teemoja, joten haastattelut lopetettiin tämän jälkeen.

Analyysin ensimmäisessä vaiheessa muodostuneet kategoriat olivat seuraavat: testausautomaatio, dokumentaatio, kehittäjälähtöisyys, ei erillistä testaajaa, projektien autonomia, itseohjautuvuus, paritestaus ja katselmointi, testauksen merkitys, osaamisen jakaminen, testaamiseen vaikuttavat tekijät, ylläpidettävyys, testaajan taidot ja ominaisuudet, asiakas, erilaiset tavat testata, vastuu laadusta, vastuu testauksesta, ympäristöt, testauksen suunnittelu sekä työkalut. Kun ensimmäiset kategoriat oli muodostettu kaikki haastatteluaineisto lajiteltiin näihin kategorioihin. Jos jokin osa aineistoa ei sopinut mihinkään kategoriaan, perustettiin uusi kategoria. Aineiston järjestelemisen jälkeen havaittiin, että osa kategorioista liittyi samaan asiaan, minkä vuoksi osa kategorioista yhdistettiin taulukon 3.2 mukaisesti.

Lopullisista kategorioista muodostettiin kustakin yksi teoria yrityksen nykyisistä testauskäytännöistä. Tämän jälkeen jokaisesta teoriasta muodostettiin yksi pykälä testauspolitiikkaan. Tavoitteena oli muodostaa jokaiseen kohtaan ohje, jota noudattamalla yrityksen parhaat testauskäytännöt toteutuvat. Testauspolitiikan muodoksi valikoitui lista ohjeita, niin sanottu kymmenisen käskyä. Testauspolitiikka järjestet-

tiin tärkeimmästä kohdasta vähiten tärkeään.

3.4. Solitan testauspolitiikka

Tässä kohdassa esitellään tutkimuksen tuloksena muodostettua testauspolitiikkaa. Testauspolitiikka on esitelty seuraavassa listassa:

§1.+2. **Huolehdi, että testaus sopii sinun projektiisi**

Huomioi ainakin projektin tavoitteet, projektimalli, koko, asiakas, kriittisyys, käyttäjät, vaikeus ja aikataulu. Mieti koskettavatko seuraavat asiat projektiasi: tietoturvatestaus, suorituskykytestaus, käytettävyystestaus, integraatiotestaus, automaattiset yksikkötestit, ympäristön testaus, hyväksymistestaus, loppukäyttäjätestaus ja regressiotestaus.

§3. **Automatisoi kaikki, minkä voit**

Automatisoi kaikki, mikä on järkevää automatisoida, myös dokumentaatio. Muista myös CI-ympäristöstä huolehtiminen.

§4. **Dokumentoi sen verran kuin on välttämätöntä**

Raportoi kaikki bugit esimerkiksi tehtävänhallintajärjestelmään, jos et korjaa niitä heti. Muista asiakkaan vaatimukset dokumentaation suhteen.

§5. **Kehittäjä, ota vastuu testauksesta, jos erillistä testaajaa ei ole**

Ilman erillistä testaajaa kehittäjillä on suuri vastuu testauksesta. Huolehdi, että testaajilla on riittävästi teknistä osaamista ja toimialuetietämystä.

§6. **Toimi asiakkaan matkaoppaana myös testaamisessa**

Opasta asiakasta testauskäytännöissä ja hyväksymistestauksessa, sekä huolehdi, että oleelliset asiat tulee testattua.

§7. **Muista, että joku joutuu ylläpitämään järjestelmää**

Laadukas koodi ja riittävä testaaminen tekevät ohjelmasta ylläpidettävämmän. Harkitse koodikatselmointikulttuuria.

§8. **Jaa (testaus)osaamista projektin sisällä ja projektien välillä**

§9. **Muista, että Solita vastaa tekemiensä järjestelmien laadusta**

Solitalla luotetaan työntekijöiden itseohjautuvuuteen ja intohimoon tehdä laadukasta koodia. Projektin sisällä projektipäällikön ja pääarkkitehdin tai -suunnittelijan on hyvä ottaa vastuu siitä, että laadusta huolehditaan.

Haastatteluiden tulokset esitellään laajemmin seuraavissa alakohdissa jaoteltuina testauspolitiikan kohtien alle.

3.4.1. Huolehdi, että testaus sopii sinun projektiisi

Yritys tekee palveluliiketoimintaa, minkä vuoksi projektit ovat erilaisia keskenään. Haastateltavien mukaan kaikki testaus suunnitellaan, toteutetaan ja dokumentoidaan projektikohtaisesti. Projektin testausta suunniteltaessa huomioidaan projektin tavoitteet, projektimalli, koko, asiakas, kriittisyys, käyttäjät, vaikeus ja aikataulu.

> "[Mikä vaikuttaa testaukseen?] Ensinnäkin onko se minkä puolen asiakkuus, onko se yksityisen puolen vai julkisen puolen, onko se iso projekti, onko se pieni, onko se perinteinen vai ketterämpi, kaikki tuollaiset vaikuttaa siihen. Yhdessä ne määritellään paljolti meilläkin asiakkaan kanssa aluksi mitä testataan, että tapauskohtaisesti menee." (Palvelupäällikkö)

Projektin tavoitteet vaikuttavat testaukseen. Osa projekteista on aikataulu- tai kustannuskriittisempiä kuin toiset, mikä näkyy projektin testauksessa. Myös projektimallilla on suuri merkitys testaukseen. Ketterämmissä projekteissa tehdään erilaista testausta kuin perinteisemmän mallin projekteissa. Myös projektin koko vaikuttaa testaukseen.

> "Projektin tavoitteet ja projektisopimukset vaikuttaa siihen [testaukseen], koska joskus saattaa olla sellaisia projekteja, joissa ei ole hirveästi rahaa tai nähdään, ettei ole aikaa käytettävissä siihen testaamiseen, ja sitten se saattaa jäädä vähän." (Ohjelmistoarkkitehti)

Asiakkaalla on myös suuri merkitys projektin testaamiseen. Eri asiakkailla on erilaiset vaatimukset laadunvarmistukselle ja asiakkaan osallistuminen testaukseen vaihtelee projektikohtaisesti. Myös se, onko asiakas julkiselta vai yksityiseltä puolelta

vaikuttaa projektin testaukseen. Projektin kriittisyys eli vaatimukset esimerkiksi järjestelmän tietoturvasta, saavutettavuudesta ja suorituskyvystä on huomioitava testauksen suunnittelussa.

> "Tekeekö Solita sitä [testausta] vai asiakas tai joku muu osapuoli, mutta semmoinen varmaan vaikuttaa. Projektimalli vaikuttaa ja projektin prioriteetti. Mutta tietysti jos on projekti semmoinen hyvin kriittinen projekti myös, pitää olla [testausta]. Projekteja on tietysti erilaisia, kriittisyydestä varmasti riippuu." (Projektipäällikkö)

Järjestelmän käyttäjät vaikuttavat myös projektin testaukseen. Osa projekteista jää asiakkaan sisäiseen käyttöön ja osa on kaikille julkisia. Myös käyttäjien määrällä ja heidän osaamistasollaan on merkitystä.

> "Millaisia loppukäyttäjät ovat, kuinka paljon käyttäjiä on, minkä monen tasoisia käyttäjiä. Kuinka paljon järjestelmän pitää olla saavutettavissa, tämän tyyppisiä asioita. Suorituskykyvaatimukset vaikuttavat ja tietoturvavaatimukset. Sitten se onko se julkinen palvelu vai sisäinen palvelu." (Projektipäällikkö)

Määrävänä tekijänä projektin testaukselle nähtiin myös projektin vaikeus. Haastateltavat totesivat, että vaikeissa projekteissa testaukseen kiinnitetään enemmän huomioita. Myös projektin aikataulu nähtiin projektin testaukseen vaikuttavana seikkana.

> "Ennakkofiilis projektin vaikeudesta vaikuttaa testauspanostuksiin." (Julkishallinnon asiantuntija)

3.4.2. Automatisoi kaikki minkä voit

Lähes kaikissa haastatteluissa tuotiin paljon esiin testauksen automatisointia. Haastateltu ohjelmistoarkkitehti kuvasi yrityksen testausprosessia seuraavasti:

> "Automatisoitu testausprosessi, se on ehkä se keskeisin juttu, mitä me tehdään monellakin tapaa. Se on itseasiassa kasvava asia, mitä enemmän meillä on jatkuvaa integraatiota, niin sitä enemmän se automaation määrä kasvaa, koska se tekee elämästä helppoa."

Myös integraatioasiantuntija ja projektipäällikkö totesivat yrityksen panostavan testiautomaatioon.

Keskeisenä asiana testiautomaatiosta haastateltavat toivat esiin jatkuvan integraation. Jatkuvalla integraatiolla tarkoitetaan koodimuutosten testaamista testitai tuotantoympäristössä usein [17]. Haastateltavat kertoivat, että projekteilla on käytössä käännöspalvelimet, jotka kääntävät ohjelmistokoodit, ajavat yksikkötestit ja julkaisevat sovelluksen säännöllisesti ja automatisoidusti. Suurimmassa osassa yrityksen projekteja on tuotantoympäristön lisäksi kehittäjien omat ympäristöt ja vähintään yksi testiympäristö.

> "Tietysti on kehittäjien omat kehitysympäristöt, ja sitten meillä on se CI-ympäristö [Continuous Integration, jatkuva integraatio] tehty Solitalle, ja sieltä on automaattisesti jatkossa päivitys asiakkaan QA-ympäristöön [Quality Assurance], joka on taas [toisen toimittajan] hallussa ja siellä on sitten myös tuotantoympäristö. Joskus tietysti voi olla vielä erikseen testiympäristö ja QA-ympäristö, mutta tässä ei nyt ole erikseen, vaan siellä on asiakkaalla vain kaksi ympäristöä. Jatkossa on tarkoitus, että tuotannosta viedään aina QA:lle ajantasaista tietoa." (Projektipäällikkö)

> "Perusmallihan on meillä se, että voit ajaa ne testit omallakin koneella, mutta yleensä ne annetaan käännöspalvelimen ajaa, hyödynnetään sitä CPU:ta. Joka ikisen commitin jälkeen käännetään ja julkaistaan ja ajetaan testit." (Ohjelmistoarkkitehti)

Automatisoitujen yksikkötestien lisäksi haastatteluissa nousi esiin tietoturvatestauksen, suorituskykytestauksen ja ympäristön testauksen mahdollinen automatisointi. Suurimmassa osassa yrityksen projekteja nämä tehdään kuitenkin käsin.

Automaattisten testien hyvänä puolena tuotiin esiin myös automaattinen statistiikka. Jokaisesta järjestelmän käännöksestä tuotetaan tietoa järjestelmän tilasta, jota voidaan käyttää myös testauksen dokumentoimiseen.

> "Automaattiset testit kirjoittaa raportin jokaisen käännöksen ja julkaisun yhteydessä, nehän on nyt itseasiassa saatu sillain, että meillä taitaa

> olla suurimmassa osassa projekteja tai ainakin uusimmissa kaksiosainen testaus. Eli on ennen julkaisua tehtävät testit, käytetään termiä yksikkötestaus, joissa on tarkoituksena, etteivät ne käy tietokannassa tai levynpinnassa tai missään muualla, ja sitten julkaisun jälkeen ajetaan savutestit, joku voisi sanoa integraatiotestaus, jolla käydään sivu kokonaan läpi ja lähetetään periaatteessa ihan HTTP-kutsuja sinne asennukseen ja katsotaan, toimiiko se asennus. Niistä jää aina jälkiä, joka kerta dokumentaatioksi käännöspalvelimelle, että miten on mennyt." (Ohjelmistoarkkitehti)

3.4.3. Dokumentoi sen verran, kuin on välttämätöntä

Haastateltavien mukaan testauksen dokumentointi on yrityksessä minimalistista. Suurella osaa projekteista on jonkinlainen testaussuunnitelma tai vähintään yhdessä sovitut testauskäytännöt dokumentoituna. Testauksen dokumentaatio on lähtökohtaisesti asiakasvaatimuksiin perustuvaa, eli dokumentaatiota tehdään sen verran kuin kukin asiakas sitä vaatii.

> "Varmaan jonkinlainen testaussuunnitelma olisi hyvä olla olemassa, projektista riippuen kuinka kattava ja missä muodossa jne." (Integraatioasiantuntija)

> "Nyt on oikeastaan yksi projekti, jossa asiakas odottaa, että hän saa testaussuunnitelman projektin lopputuloksena, niin meidän täytyy ratkoa, kuinka kevyt me siitä tehdään, ja miten me se tehdään, miten me kuvataan, että miten me ollaan testattu kaikkia asioita ja osavaiheita." (Ohjelmistoarkkitehti)

Yrityksessä käytetään suurimmassa osassa projekteja apuna JIRA-tehtävänhallintaohjelmistoa [18]. JIRA:an on kirjattu toteutustehtävät, ja yleensä testauksen havainnot kirjataan JIRA:an joko siihen liittyvälle tehtävälle tai niistä tehdään oma tehtävä. Kuitenkin esimerkiksi suorituskykytestauksista tai muista isojen kokonaisuuksien testauksista voidaan kirjoittaa laajempiakin raportteja.

"Havaintojen seuranta eli jonkinlainen tikettijärjestelmä, jira esimerkiksi, mihin kerätään ylös havaintoja ja missä niitä pystytään pistämään tehtävälistalle" (Integraatioasiantuntija)

"Suorituskykytestauksesta, kun se ajetaan yleensä ennen julkaisua projekteissa, niin niistä yleensä kirjoitetaan käsin raporttia. Se voi olla JIRA-tiketti, johon kommentoidaan, tai tehdään erikseen Word-dokumentti, johon kirjataan, millaisia testejä on ajettu, millekin urleille, ja miten sovellus on toiminut millaisellakin kuormalla, ja arvioidaan sen pohjalta jonkinlainen loppupäätelmä, että tämä kestää tällaista kuormaa tällaisessa tilanteessa." (Ohjelmistoarkkitehti)

3.4.4. Kehittäjä, ota vastuu testauksesta, jos erillistä testaajaa ei ole

Suurimmassa osassa yrityksen projekteja ei ole erillistä vastuuhenkilöä testaukselle. Tämä tarkoittaa käytännössä sitä, että ohjelmistokehittäjät vastaavat myös testauksesta suurilta osin. Kehittäjien vastuulle jää suurin osa kehityksen aikaisesta testauksesta. Tämän jälkeen asiakas suorittaa hyväksymistestauksen. Välillä myös projekti- tai palvelupäällikkö saattaa osallistua järjestelmätestaukseen.

Monitoimittajaprojekteissa voidaan tarvittaessa järjestää testauspäiviä, joihin osallistuu useampi toimittaja ja mahdollisuuksien mukaan asiakas. Nämä testauspäivät vaativat enemmän suunnittelua, josta vastaa yleensä projektin arkkitehti.

Kun yritykseen palkataan uusia työntekijöitä, tavoitteena on palkata henkilöitä, jotka pystyvät ottamaan vastuun myös testauksesta. Haastatellun arkkitehdin mukaan myös testauksen tulee lähteä työntekijöiden ammattitaidosta ja intohimosta tehdä laadukkaita järjestelmiä.

"Pitää lähteä niistä ihmisistä, tekijöistä ja niiden ammattitaidosta ja intohimosta, että haluaa tehdä sitä parasta." (Ohjelmistoarkkitehti)

"Koitetaan palkata ihmisiä, jotka osaavat päätellä asioita itse" (Ohjelmistoarkkitehti)

3.4.5. Toimi asiakkaan matkaoppaana myös testaamisessa

Haastateltavien mukaan asiakas on isossa roolissa testaamisessa, sillä usein vain asiakkaalla on riittävä toimialuetietämys kehitettävästä järjestelmästä.

> "Ainakin tuossa ylläpidossa, juuri se, että asiakas on viime kädessä silleen vastuullinen, että mitä sinne tuotantoon menee, koska he antaa sen lopullisen kuittauksen, eli heillä on tosi tärkeä rooli, mutta he ei välttämättä itse sitä aina tiedä, kuinka tärkeä rooli heillä on siinä." (Palvelupäällikkö)

> "Jos se [testaaja] on taas asiakkaan henkilö, niin sehän tietää sitten hyvin tarkkaan, miten asioiden pitäisi toimia." (Ohjelmistoarkkitehti)

Lisäksi useimmissa projekteissa ennen luovutusta tehdään hyväksymistestaus, jossa asiakas varmistaa järjestelmän toiminnan. Kuitenkaan haastateltavien kokemuksen mukaan asiakkaalta ei löydy aina tarvittavaa testausosaamista ja tietämystä järjestelmän teknisistä ominaisuuksista. Tälläisissa tilanteissa yritys on tarjonnut asiakkaalle konsultaatiota ja tukea testauksen suorittamiseen sekä suunnitteluun.

> "Sitten sellaista opastusta joka tulee suorituskykytestaamisen ja tietoturvatestaamiseen, niin sitähän me tehdään asiakkaille, että tämä vaatii tämän ja tässä pitäisi tällainen asia tehdä tai miksi tämä on hyvä asia tehdä." (Ohjelmistoarkkitehti)

> "Jos se [testauksen suunnittelu] on kauhean asiakasvetoista, keskimäärin silloin ei minun mielestä kauhean hyvin toteudu. Jos me ollaan siinä enemmän ohjaamassa sitä ja mielellään jos se on vielä joku muu kuin se, joka on sen ominaisuuden kehittäjä, niin kyllä se sitten tulee yleensä organisoitua paremmin." (Integraatioasiantuntija)

3.4.6. Muista, että joku joutuu ylläpitämään järjestelmää

Tutkimukseen osallistunut palvelupäällikkö toi ilmi, että ylläpitovaiheessa joudutaan usein palaamaan asioihin, joihin on jäänyt virheitä tai jotka eivät ole täysin vastanneet määrittelyä.

> "Ja usein kuitenkin palataan niihin asioihin, jotka on projektin aikana tehty väärällä lailla tai jätetty tekemättä tai ei olla testauksessa huomattu ja ylläpidossa kiistellään onko tämä takuutöitä vai eikö ole. Niin varmaan vielä huolellisempi ja tarkempi testaus, ja asiakkaan sitouttaminen myös siihen, että se jos asiakas tämän hyväksyy, niin se on kanssa hyväksytty eikä tarvitsisi ylläpidossa palata niihin niin paljon, niitä jonkin verran on." (Palvelupäällikkö)

Myös myyntiasiantuntija ja integraatioasiantuntija toivat esiin, että projektien kustannukset vähenevät ylläpitovaiheessa kunnollisella testauksella. Haastatellun ohjelmistoarkkitehdin mukaan ohjelmakoodin testattavuus ja testit sekä niiden automatisointi tekevät ohjelmasta paremmin ylläpidettävän. Testattava ohjelmisto kannustaa automaattisten ja manuaalisten testien olemassaoloon [19]. Testattavuuden tärkeimmät osa-alueet on tarkkailtavuus ja hallittavuus [19].

> "Kun taas koetaan joku asia vaikeaksi niin siihen kirjoitetaankin sellaista koodia mikä on testattavaa, niin sen [testauksen] pystyy tekemään tosi pienellä vaivalla ja se kehittäjä pystyy oikeasti tekemään sen toiminnallisuuden nopeammin ja lopulta sitten sovelluksesta tulee ylläpidettävämpi monessakin suhteessa: sen koodi on ylläpidettävämpää, siinä on testejä ja se on sen takia esimerkiksi ylläpidettävämpää." (Ohjelmistoarkkitehti)

Yrityksessä suositaan jonkin verran myös koodikatselmointia ja ristiin testaamista. Kun yksi kehittäjä saa ominaisuuden valmiiksi, joku toinen kehittäjä käy toteuksen läpi. Näin varmistetaan, että toteus on tehty järkevästi, ja löydetään mahdollisia vikoja. Koodikatselmoinnissa hyödynnetään myös työkaluja, jotka löytävät ohjelmakoodista ongelmakohtia.

> "Esimerkiksi koodikatselmoinnit jotka liittyvät laadunvarmistukseen. Siitähän me ei olla vielä keskusteltu ollenkaan, pienissä projekteissa vähemmän mutta nyt isommissa projekteissa enemmän ja enemmän koodikatselmointikäytäntöjä." (Ohjelmistoarkkitehti)

> "En tiedä mihin kysymykseen liittyy, tuli vain mieleen, että ollaan jonkin verran tehty parikatselmointia. Yhdessäkin projektissa sovittu, että

> ennen kuin se on valmis niin pitää käydä jonkun toisen silmien läpi vielä, että saa siihen kommentit. Ei lähinnä, että löytäisi sieltä vikoja vaan se että onko se järkevästi tehtyä, sehän on osittain sitä laatua sitten kumminkin." (Integraatioasiantuntija)

> "Ehkä sitten tehdään paritestausta myös jonkin verran, että kun joku on valmis kehittäjän mielestä niin joku toinen varmistaa, joko testaamalla tai katselmoimalla koodin tai jollain muulla tavalla." (Projektipäällikkö)

3.4.7. Jaa (testaus)osaamista projektin sisällä ja projektien välillä

Yrityksessä on töissä vain muutama testausasiantuntija, minkä vuoksi lähes kaikki joutuvat osallistumaan testaukseen. Haastateltavat kertoivat, että lähes kaikki projektissa ovat jollain tavoin osallisena testauksessa. Tämän vuoksi testaukseen liittyvää tietämystä on jaettava. Yrityksessä käytetään osaamisen jakamisen keinoina tietoiskuja ja osaamisyhteisöjä. Tietoiskut ovat noin 30 minuutin mittaisia esitelmiä esittelijän valitsemasta aiheesta. Usein aiheet liittyvät ratkaisuihin, joita on tehty projekteissa. Osaamisyhteisöt puolestaan ovat jonkin tietyn teknologian tai osaamisalueen (esim. Java tai testaaminen) ympärille keskittyneitä ryhmiä, jotka keskustelevat ja jakavat tietoa aiheeseen liittyen. Yrityksessä on järjestetty testausaiheisia tietoiskuja ja on myös testausosaamisyhteisö. Osaamisyhteisöiltä voi myös kysyä apua kyseisen aihealueen ongelmiin.

> "Eli ihmiset katsovat omia tarpeitaan ja asioitaan ja kehittävät sitä siinä projektityön ohessa, ja sitä kautta tulee kaikki uudet hyvät tavat ja jutut, jotka sitten valuvat osaamisen jakamisen kautta toisiin projekteihin." (Ohjelmistoarkkitehti)

Lisäksi haastateltavat toivat esiin, että yleensä projektitiimistä löytyy joku, jolla on testausosaamista. Osaamista pyritään jakamaan myös projektin sisällä. Sen sijaan, että yksi henkilö tekisi aina esimerkiksi suorituskykytestauksen, voi tämä henkilö opastaa muita.

> "Osaamista pitää kuitenkin jakaa ja ihmisiä valistaa. Jos ne ei tiedä jostain asiasta, niin ne opetetaan tekemään se asia sen sijaan että laitettaisiin se tekeminen 100 prosenttisesti jollekin yhdelle yksittäiselle henkilölle. Mitä enemmän meillä on ihmisiä, jotka osaa tehdä sen ja ajatella sen, niin sitä enempi meillä kehittyy se maailma ja meillä on aina fiksumpia ihmisiä talossa jotka osaa ajatella laaja-alaisemmin." (Ohjelmistoarkkitehti)

Osa haastateltavista mainitsi ensin kysyvänsä apua vertaisiltaan, esimerkiksi palvelupäällikkö palvelupäälliköltä tai kehittäjä kehittäjältä. Osa kertoi kysyvänsä apua yrityksen testausasiantuntijoilta.

3.4.8. Muista, että Solita vastaa tekemiensä järjestelmien laadusta

Haastateltavien mukaan yritys itse vastaa työnsä laadusta. Vaikka vastuut ovat yleensä kirjattuna sopimuksiin, haastateltavat kokivat, että tehdyn työn laadusta vastataan itse. Kuitenkin vastuu on osittain myös muiden sidosryhmien, erityisesti asiakkaan, vastuulla.

> "Solitan vastuulla, kun tehdään semmoista kokonaistoimitusta" (Julkishallinnon asiantuntija)

> "Käytännössähän se on aina tekijöiden vastuulla se sovelluksen toteutuksellinen laatu." (Ohjelmistoarkkitehti)

> "Kyllä minä näkisin, että se on kuitenkin meidän vastuulla toimittajana." (Palvelupäällikkö)

Yrityksen sisällä vastuu laadusta kuuluu haastateltavien mukaan kaikille projektissa oleville. Kuitenkin projektipäälliköllä ja pääsuunnitelijalla tai ohjelmistoarkkitehdilla koetaan olevan suurempi vastuu.

> "Jokaisen vastuullahan se on. Viime kädessä vastaa projektipäällikkö projektin puitteissa. Pääsuunnittelijalla/arkkitehdillä on jonkinlainen vastuu." (Integraatioasiantuntija)

4. TESTAUSPOLITIIKAN SOVELTAMINEN CASE-PROJEKTISSA

Tässä luvussa käsitellään testauspolitiikan soveltamista case-projektiin. Projektille kehitettiin yhdessä projektitiimin kanssa testauspolitiikkaan pohjautuva testausstrategia. Projektia seurattiin sen aikana ja seurannan lopuksi projektitiimiä haastateltiin.

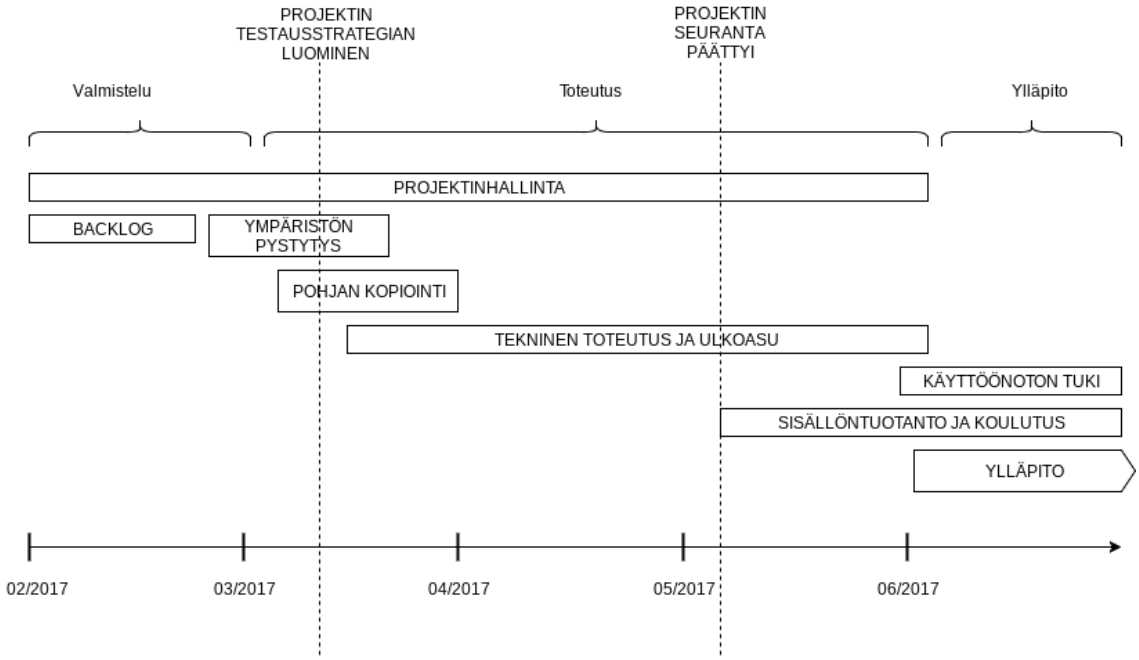
Kohdassa 4.1 esitellään case-projekti ja projektitiimi. Kohta 4.2 selvittää, miten projektille luotiin testausstrategia, ja kohta 4.3 esittelee luodun testausstrategian. Projetin aikana suoritettu seuranta ja sen tulokset esitellään kohdassa 4.4. Projektitiimin haastattelut ja niiden tulokset kuvataan luvun lopussa kohdassa 4.5.

4.1. Case-projektin kuvaus ja rajaus

Tässä diplomityössä esimerkkiprojektina toimii yrityksen julkisen puolen asiakkaan verkkosivuston uudistaminen. Uuden sivuston toteutus perustuu Episerver-sisällönhallintajärjestelmään [20]. Uusi verkkosivusto perustuu toisen julkisen puolen asiakkaan sivustolta saatavaan toteutuspohjaan. Tämä tarkoittaa sitä, että sivustolle tulevien sivupohjien ja sisältötyyppien toteutus kopioitiin aiemmin tehdystä projektista. Suurin osa tehtävästä toteutustyöstä keskittyi verkkosivuston ulkoasun muuntamiseen tämän asiakkaan toivomaksi. Projektin aikataulu ja rajaus on esitelty alakohdassa 4.1.1. Projektitiimi on kuvattu alakohdassa 4.1.2.

4.1.1. Projektin aikataulu ja rajaus

Projektin kesto oli tarjousvaiheesta käyttöönottoon noin puoli vuotta. Projektin aikataulu on esitetty kuvassa 4.1. Projekti alkoi tarjouksen lähettämisestä asiakkaalle projektisuunnitelman kera. Projektisuunnitelmassa sivuttiin myös testausta ja laadunvarmistusta yleisiltä reunaehdoiltaan. Tämän suunnitelman perusteella asiakas



Kuva 4.1: Caseprojektin aikataulu

hyväksyi toteutustarjouksen. Kun asiakas hyväksyi tämän, seurasi projektin kehitysjonon (backlog) suunnitteleminen yhdessä asiakkaan ja mainostoimiston edustajien kanssa. Projektin suunnitteluvaiheessa projektiin ei oltu vielä resursoitu käyttöliittymäpuolen kehittäjää.

Kun alustava kehitysjono oli muodostettu, aloitettiin projekin toteutusvaihe. Aluksi pystytettiin kehitysympäristö, käännöspalvelimet ja testiympäristöt. Tämän jälkeen kopioitiin toisesta projektista toteutus pohjaksi. Tässä kohtaa muodostettiin myös yhdessä projektitiimin kanssa projektin testausstrategia.

Pohjan kopioimisen jälkeen projektissa aloitettiin kehitystyö. Tämä jatkui toukokuun alkupuolelle, jolloin asiakas aloitti testaamisen. Asiakkaan testaamisen aikana havaitsemia virheitä korjattiin niiden löytyessä. Käyttöönotto on suunniteltu kesäkuun alkupuolelle. Käyttöönoton jälkeen alkaa projektin ylläpitovaihe.

Projektin seuranta rajoitettiin vain osaan projektia. Projektia seurattiin sen alusta kehitystyön loppuun. Tämä sisälsi projektin suunnitteluvaiheen, jossa muodostettiin projektin projektisuunnitelma. Seuranta lopetettiin ennen kuin asiakas aloitti testaamisen. Näin seuranta kohdistui pelkästään kohdeyrityksen omiin testausaktiviteetteihin.

4.1.2. Projektitiimi

Projektitiimi koostui yrityksen puolelta projektipäälliköstä, ohjelmistoarkkitehdista ja kahdesta ohjelmistokehittäjästä. Projektin arkkitehti suunnitteli järjestelmän toteutuksen ja kopioi pohjan aiemmasta projektista. Toinen ohjelmistokehittäjistä keskittyi enemmän sivuston palvelinpään toteutuksiin ja toinen sivuston ulkoasuun. Ulkoasuun keskittyvä kehittäjä ei työskennellyt kohdeyrityksessä vaan oli ostettu projektiin alihankkijana toisesta yrityksestä.

Asiakkaan puolelta projektissa oli mukana heidän projektipäällikkönsä sekä sivuston sisällöntuottajia. Lisäksi projektissa oli mukana kolmantena osapuolena mainostoimisto, joka vastasi sivuston visuaalisesta suunnittelusta ja konseptoinnista.

Projektia johtanut projektipäällikkö oli tutkimuksen aikana työskennellyt kohdeyrityksessä noin 3,5 vuotta. Hän oli ehtinyt ennen kyseistä projektia työskennellä kohdeyrityksessä kuudessa eri web-projektissa. Yhteensä työuransa aikana projektipäällikkö oli ehtinyt työskennellä noin kolmessakymmenessä projektissa. Tämän projektin aikana hänen työtehtävänsä sisälsivät suunnittelua, tehtäväjonon hallintaa, projektitiimin ohjausta, asiakkaan ohjausta, viestintää, budjetin ja aikataulun hallintaa sekä välillä myös testausta.

Projektin arkkitehtuurista vastannut henkilö oli tutkimuksen aikana työskennellyt noin 3,5 vuotta kohdeyrityksessä. Hänellä oli kokemusta noin kahdestatoista samalla teknologialla toteutetusta projektista kohdeyrityksessä. Lisäksi hän oli ennen kohdeyrityksessä työskentelyä ollut lukuisissa samankaltaisissa projekteissa muissa yrityksissä. Hänen tehtäviinsä kuului järjestelmän arkkitehtuurin suunnittelua, ohjelmakoodin laadusta huolehtimista, asiakkaan kouluttamista, kehittämistä, testaamista ja muita teknisiä tehtäviä kuten CI-ympäristön pystyttämistä.

Toinen projektin kehittäjistä keskittyi projektin aikana enemmän palvelinpään toteutuksiin. Hän oli projektin aikana työskennellyt kohdeyrityksessä noin 7 vuotta. Tänä aikana hän oli ehtinyt olla mukana noin 20 eri projektissa. Tämä oli kuitenkin hänen ensimmäinen projektinsa projektissa käytetyllä teknologialla. Hänen tehtäviinsä kuului ohjelmointia, testausta ja ympäristöistä huolehtimista.

Järjestelmän ulkoasusta vastannut kehittäjä ei työskennellyt kohdeyrityksessä, vaan hän oli alihankkija toisesta yrityksestä. Tämä oli hänen ensimmäinen projek-

tinsa kohdeyrityksessä ja myös käytetyllä teknologialla. Hän oli kuitenkin työskennellyt ennen tätä kymmennissä web-kehitysprojekteissa. Projektin aikana kehittäjän tehtäviin kuului sivuston visuaalisen ilmeen toteuttaminen ja testaaminen.

4.2. Testausstrategian muodostaminen

Projektin tehtävälistan muodostumisen jälkeen pidettiin kohdeyrityksen projektitiimin kanssa testaussuunnittelutyöpaja, jossa projektisuunnitelman tavoitteita peilattiin testauspolitiikkaan ja sitä kautta muodostettiin testausstrategia. Työpajassa käytiin testauspolitiikka läpi pykälä pykälältä ja selvitettiin jokaiseen pykälään liittyvät seikat. Työpajassa läpikäydyt asiat ovat esiteltyinä tarkemmin alakohdissa 4.2.1-4.2.8.

4.2.1. Huolehdi, että testaus sopii sinun projektiisi

Testauspolitiikan ensimmmäisten pykälien aikana selvitettiin projektin tavoitteita, projektimallia, kokoa, asiakasta, kriittisyyttä, käyttäjiä, vaikeutta ja aikataulua ja pohdittiin kuinka ne tulee huomioida testauksessa. Tässä kohdassa käytiin myös läpi projektin tarpeet tietoturvatestauksen, suorituskykytestauksen, käytettävyystestauksen, integraatiotestauksen, automaattitestien, ympäristöjen testauksen, hyväksymistestauksen, loppukäyttäjätestauksen ja regressiotestauksen osalta.

Projektille, jonka pohjalta tämä projekti toteutetaan on tehty tietoturvatestausta. Tämän katsottiin riittävän myös tämän projektin tietoturvatestaukseksi, koska järjestelmä on lähinnä informaation esittämistä verkkosivuilla.

Järjestelmällä ei ole suorituskykyvaatimuksia, eivätkä järjestelmän kävijämäärät tule olemaan suuria. Lisäksi ei ole kriittistä, vaikka järjestelmä olisi hetkellisesti saavuttamattomissa. Tämän vuoksi myöskään suorituskykytestausta ei tehdä tämän projektin puitteissa.

Sivuston konsepti ja ulkoasu tulevat kolmannelta osapuolelta. Tämän vuoksi yrityksen vastuulle ei kuulu järjestelmän käytettävyys. Konseptin suunnitteleva mainostoimisto saattaa tehdä käytettävyystestausta tai loppukäyttäjätestausta, mutta toteutusprojektin laajuuteen ne eivät kuulu.

Järjestelmään ei tule integraatioita, joten integraatiotestausta ei tarvitse huo-

mioida. Ympäristöjen testausta ei tässä projektissa tehdä, sillä ympäristöt luodaan osin käsin käyttäen pilvipalvelun valmiita pohjia.

Regressiotestausta sovittiin tehtäväksi uusien versioiden kohdalla. Regressiotestauksessa päätettiin testata tärkeimmät ominaisuudet manuaalisesti eli etusivun latautuminen, uuden sivun luominen, uuden artikkelin luominen ja lohkojen nostaminen sivuille.

4.2.2. Automatisoi kaikki, minkä voit

Testauspolitiikan kolmantena pykälänä on testauksen automatisointi. Tämän pykälän yhteydessä käytiin läpi, kuinka projektin testaus automatisoidaan. Lisäksi käsiteltiin ympäristöihin liittyvät asiat eli CI-palvelin, testiympäristö ja tuotantoympäristö. Tämän yhteydessä sovittiin myös automatisointiin käytettävistä työkaluista ja teknologioista. Lopuksi keskusteltiin siitä, minkälaisia automaattisia testejä tehdään.

Projektilla on käytössään Jenkins CI-palvelin [21]. Jokainen versionhallintaan lisätty koodimuutos käännetään välittömästi CI-palvelimella. Jos käännös epäonnistuu tai automaattiset testit eivät mene läpi, lähetetään siitä ilmoitus projektin keskustelukanavalle. Joka yö CI-palvelin päivittää testiympäristöön uuden version. Myös tuotantoasennukset tehdään CI-palvelimen kautta.

Projektin ollessa ulkoasukeskeinen ja pohjautuessa sisällönhallintajärjestelmään järkevien automaattisten testien määrä jää vähäiseksi. Jokaisen käännöksen yhteydessä testataan, että sivupuun sivuille pääsee ja ettei niistä tule JavaScript-virheitä.

4.2.3. Dokumentoi sen verran, kuin on välttämätöntä

Testausstrategiaan selvitettiin projektin dokumentaation tarpeet. Kohdassa käytiin läpi asiakkaan vaatimukset dokumentaatiolle. Lisäksi suunniteltiin käytännöt testaushavaintojen raportoinnille ja tehtävien työnkululle.

Asiakkaan puolelta ei ole vaatimuksia testauksen dokumentaatiolle. Projektissa käytetään Kanban-taulua, jossa on kolme tilaa: to do, in progress ja done. Kanban-taulu on työkalu työnkulun visualisointiin [22]. Kun tehtävä on kehittäjän mielestä valmis, hän siirtää sen done-tilaan ja kommentoi tehtävälle miten sitä on testattu.

Sisäisesti tehdyt huomiot testauksesta kirjataan kommentteina tehtäville. Asiakkaan huomioista tehdään uusi tehtävä. Huomioita voidaan korjata myös heti, kun niitä huomataan, jolloin niitä ei tarvitse kirjata ylös.

4.2.4. Ilman erillistä testaajaa, kehittäjillä on suuri vastuu

Tässä pykälässä käytiin läpi, ketkä testaavat järjestelmää. Lisäksi sovittiin toimittajan ja asiakkaan vastuista ja siitä, mitä pitää olla testattuna ennen asiakkaan hyväksymistestausta.

Projektissa testausta tekevät kehittäjät, projektipäällikkö ja asiakas. Projektin aikana kehitystä tekevät pääosin kehittäjät ja projektipäällikkö. Kun kehittäjä on saanut tehtävän valmiiksi, joku muu katsoo sen läpi. Tätä varten projektissa järjestetään noin kahden viikon välein testaussessioita, joissa kehittäjät ja projektipäällikkö keskittyvät valmistuneiden tehtävien testaamiseen.

4.2.5. Toimi asiakkaan matkaoppaana myös testaamisessa

Tämän pykälän kohdalla suunniteltiin, kuinka asiakasta tuetaan testauksessa ja kuinka asiakkaan toimialuetietämys saadaan hyödynnettyä. Tarkempi suunnittelu liittyen asiakkaan testaukseen sovittiin tehtäväksi projektin aikana.

Asiakas testaa järjestelmää samalla kuin aloittaa sisällöntuoton järjestelmään. Suurimmalla osalla asiakkaan sisällöntuottajia ei ole kokemusta Episerver-julkaisujärjestelmästä. Tästä johtuen heille järjestetään sisällöntuottokoulutusta, jossa testataan heidän kanssaan yhdessä. Asiakkaalle annetaan testauksen tueksi käyttöohjeita ja testausohjeita. Testausohjeissa hyödynnetään kehittäjien tehtäville kirjaamia testausdokumentaatioita.

4.2.6. Muista, että joku joutuu ylläpitämään järjestelmää

Testausstrategiaan kirjattiin ylös myös, kuinka järjestelmän ylläpidettävyydestä huolehditaan. Tämän yhteydessä keskusteltiin sovelluskoodin testattavuudesta ja ylläpidettävyydestä ja sovittiin koodikatselmointikäytännöistä.

Projektissa sovittiin, että koodikatselmointeja pidetään varsinkin alussa säännöllisesti. Katselmoinnista vastaa projektin arkkitehti, mutta kehittäjät ovat itse

proaktiivisia ja pyytävät arkkitehdilta katselmointeja.

4.2.7. Jaa (testaus)osaamista projektin sisällä ja projektien välillä

Tämän pykälän yhteydessä keskusteltiin siitä, kuinka projekti voi hyödyntää muuta talosta löytyvää testausosaamista projektissa. Tämän yhteydessä keskusteltiin myös projektitiimiläisten omasta osaamisesta, ja miten sitä voidaan hyödyntää ja jakaa projektin sisällä.

Projektin kehittäjät ovat osallistuneet myös muihin samankaltaisten järjestelmien kehittämiseen ja oppineet niistä hyviä testauskäytäntöjä. Lisäksi tämän projektin testauskäytäntöjä viedään muihin projekteihin, jos ne todetaan hyviksi.

4.2.8. Solita vastaa tekemiensä järjestelmien laadusta

Tässä pykälässä käytiin läpi projektin laatuvaatimuksia ja sopimusvelvoitteita. Tämän lisäksi sovittiin siitä, kuka vastaa projektin laadusta projektin sisällä.

Todettiin, että vaikka järjestelmä ei ole laatukriittinen, on asiakas yritykselle tärkeä. Tämän vuoksi pyritään tekemään hyvää laatua budjetti huomioiden. Projekti on pieni ja myynnissä ei ole huomioitu laadunvarmistusta kovin laajassa mittakaavassa, minkä vuoksi testauspanokset on käytettävä järkevästi.

Projektipäällikkö vastaa projektin laadusta yrityksen sisällä. Projektipäällikkö varmistaa myös asiakkaan kanssa, mitkä asiat on sovittu toteutettavaksi projektin puitteissa ja mitkä huomiot jäävät projektin laajuuden ulkopuolelle.

4.3. Case-projektin testausstrategia

Case-projekti tarvitsi mahdollisimman kevyen testausstrategian projektin pienen koon ja vähäisten dokumentaatiovaatimusten vuoksi. Testausstrategia päädyttiin kirjaamaan ylös projektin wiki-sivulle vapaamuotoisesti. Testauksen tasoja projektille ei määritelty, sillä projektitiimi ei kokenut niitä tarpeellisiksi. Käsitellyt asiat jaoteltiin seuraavien otsikoiden alle: erilaiset testaukset, testiautomaatio, dokumentaatio, testaajat, koodikatselmointi, laatuvaatimukset ja asiakas. Jaottelu testaus-

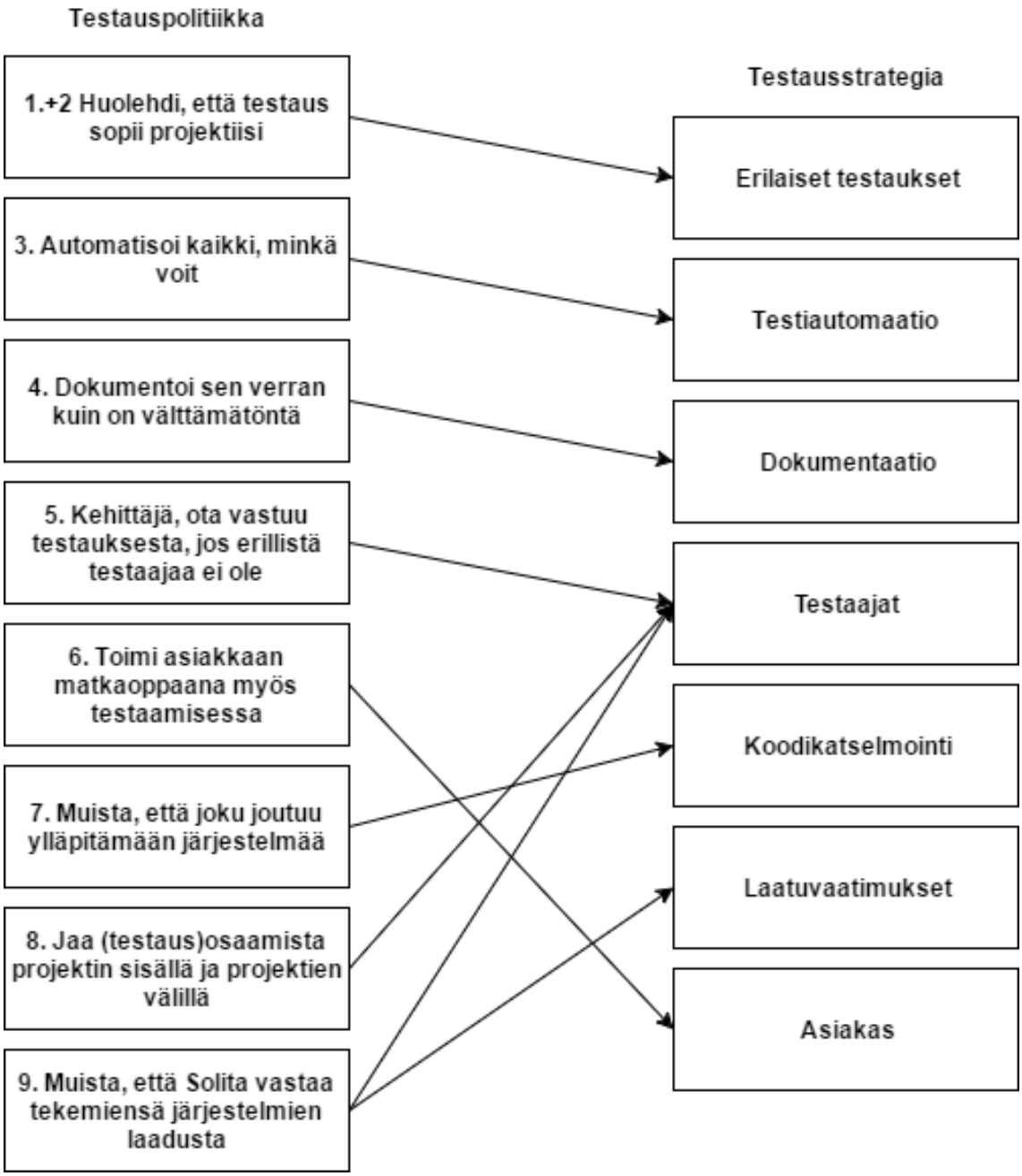
politiikasta testausstrategiaksi on esitelty kuvassa 4.2 ja testausstrategia kokonaisuudessaan on kuvassa 4.3.

Erilaiset testaukset -osion alle kirjattiin asiat, joista keskusteltiin testauspolitiikan pykälän *huolehdi, että testaus sopii projektiisi* yhteydessä. Testiautomaatio-osio käsittää kaikki testauksen automatisointiin liittyvät asiat, eli lähinnä testauspolitiikan pykälän *automatisoi kaikki minkä voit* huomiot. Dokumentaation alle listattiin *dokumentoi sen verran, kuin on välttämätöntä* -pykälän huomiot. Testaajat-otsikon alle kirjattiin *kehittäjä, ota vastuu testauksesta, jos erillistä testaajaa ei ole* -pykälän aikana keskustellut asiat. Koodikatselmointiin listattiin testauspoliitikan 6. ylläpitoa käsittelevän pykälän asiat. Laatuvaatimuksiin kirjattiin muistiin asiat, joista keskusteltiin testauspolitiikan pykälän *muista, että Solita vastaa tekemiensä järjestelmien laadusta* aikana. Asiakas kohtaan kirjattiin *toimi asiakkaan matkaoppaana myös testaamisessa* -pykälässä keskustellut asiat ja hyväksymistestaukseen liittyvät seikat.

4.4. Projektin aikainen seuranta ja sen tulokset

Heti projektin testausstrategian muodostamisen jälkeen projektitiimille laitettiin lyhyt kysely sähköisesti siitä, kuinka hyödylliseksi he kokivat testauspolitiikan strategian muodostamisessa. Kysely toteutettiin Google Formsin avulla [23]. Lisäksi kysyttiin, oliko testauspolitiikka riittävän kattava projektin testausstrategian muodostamiseen ja oliko heidän mielestään testauspolitiikka ylipäänsä yrityksen toimintaan sopiva. Kyselyn tulokset esitellään alakohdassa 4.4.1.

Projektin aikana seurattiin sovittujen käytäntöjen toteutumista. Seurantaa suoritettiin tehtävänhallintajärjestelmästä, projektin wiki-sivulta ja CI-palvelimen lokeista. Tarkkailuissa kiinnitettiin huomioita sovittujen työnkulkukäytäntöjen toteutumiseen, testausaikataulun noudattamiseen ja automaattitesteihin, sekä niiden hyödyntämiseen. Projektitiimi piti viikottaisiksi suunniteltuja tapaamisia, joissa käytiin läpi, mitä kukakin on tehnyt ja miten työt etenevät. Myös näissä tapauksissa seurattiin testauspanosten edistymistä. Projektin aikaisen seurannan havainnot esitellään alakohdassa 4.4.2.



Kuva 4.2: Jaottelu testauspolitiikasta testausstrategiaksi

Erilaiset testaukset

Projekti on pieni ja projektissa ei ole kriittistä logiikkaa tai toiminallisuuksia. Lisäksi projekti pohjautuu aiempaan projektiin, jonka puitteissa testausta on tehty.

Suorituskykytestaus: Ei ole suorituskykyvaatimuksia, suorituskykyä testattu projektissa, johon tämä pohjautuu. Sivuston käyttäjämäärät pieniä.

Tietoturvatestaus: Ei tietoturvakriittinen, testattu projektissa, johon tämä pohjautuu.

Käytettävyystestaus: Konsepti ja ulkoasu tulevat mainostoimistolta, jolloin käytettävyys myös heidän vastuullaan.

Ympäristön testaus: Azurelta valmiita ympäristöjä eli on testattu sen puitteissa, asennetaan käsin eli ei testattavia asennus-skriptejä.

Hyväksymistestaus: Asiakas tekee hyväksymistestauksen sisällöntuoton yhteydessä.

Regressiotestaus: Kun päivitetään uusia versioita tarkistetaan, että tärkeimmät ominaisuudet toimivat: artikkelin luonti, lohkon nostaminen sivulle, etusivun latautuminen.

Jokaisen asennuksen yhteydessä automaattiset testit tarkastavat sivupuusta kaikki sivut, ettei niistä tule javascript-virheitä.

Loppukäyttäjätestaus: Ei meidän toimesta, mainostoimisto saattaa tehdä.

Esteettömyys: Mainostoimiston vastuulle.

Testiautomaatio

CI-palvelin päivittää testiympäristön joka yö. Jokainen commit ajetaan CI-palvelimella ja samalla ajetaan automaattiset testit. CI-palvelimelle jää logit testeistä. Lisäksi CI-palvelin hälyttää projektiryhmän chattiin, jos tulee virheitä. Samoin tuotannonvirheistä lähetetään sähköpostihälytykset.

Dokumentaatio

Asiakkaalta ei ole vaatimuksia testauksen dokumentaatiosta. Sisäisesti huomiot kirjataan tiketille ja asiakkaan huomioista tehdään omat tiketit. Kanban-boardissa kolme tilaa: todo, in progress ja done.

Kun tiketti on valmistunut, kehittäjä kirjaa tiketille lyhyesti, kuinka sitä on testannut.

Testaajat

Kun ominaisuus on kehittäjän mielestä valmis, joku muu (esim. projektipäällikkö) varmistaa vielä toiminnallisuuden. Järjestetään testaussessioita, joissa käydään yhdessä läpi muiden tuotoksia. Testaukseen osallistuvat kehittäjät, projektipäällikkö ja asiakas.

Koodikatselmointi

Koska pieni tiimi, ei oteta käyttöön pullrequestaja. Projektin arkkitehti huolehtii koodin katselmoinnista säännöllisesti varsinkin alkuvaiheessa. Kehittäjät ovat proaktiivisia ja pyytävät arkkitehtiä tekemään katselmointeja.

Laatuvaatimukset

Projektilla ei ole tiukkoja laatuvaatimuksia, mutta halutaan näkyä asiakkaalle hyvänä ja laadukkaana toimijana, joten panostetaan riittävästi laatuun.

Asiakas

Asiakas tarvitsee tukea ja koulutusta testaukseen. Testataan aluksi yhdessä asiakkaan kanssa, jotta he pääsevät alkuun sisällöntuoton kanssa. Annetaan asiakkaalle myös testausohjeita. Asiakkaan testausta tarvitaan sisällöntuoton yhteydessä, koska osaa asioista ei voi tietää ennen kuin sisältöä aletaan luomaan.

Sovitaan asiakkaan kanssa asiat, jotka tehdään tämän projektin puitteissa ja hyväksytetään, jottei ylläpitovaiheessa jää epäselvyyksiä.

Kuva 4.3: Projektin testausstrategia

4.4.1. Kyselyn tulokset

Testausstrategian luomisen jälkeen projektitiimille lähetetty kysely ei onnistunut. Vain yksi henkilö projektitiimistä vastasi kyselyyn ennen loppuhaastatteluja muistutteluista huolimatta. Tämäkin tapahtui noin kaksi viikkoa strategian luomisen jälkeen.

Haastatteluiden jälkeen ja 1,5 kuukautta testausstrategian luomisen jälkeen annetut vastaukset eivät vastaa enää kyselyltä toivottuun käyttötarkoitukseen. Tavoitteena oli saada nopeasti palautetta, kun testausstrategian luominen on vielä lähimuistissa. Myös haastatteluissa käydyt keskustelut vaikuttavat kyselyn tuloksiin, eikä kysely rajaudu siinä kohtaa enää pelkästään kyseiseen projektiin ja kyseiseen tilaisuuteen. Tämän vuoksi kyselystä saatuja tuloksia ei hyödynnetty tässä tutkimuksessa.

4.4.2. Seurannan tulokset

Jo projektin seurannan alkuvaiheilla kävi ilmi, että projektimalli ei pysynytkään suunitellussa. Projekti ei oikeastaan hyödyntänyt tehtävänhallintajärjestelmää, vaan kehitystehtävät jaettiin projektin sisällä vapaamuotoisesti. Tehtävät pohjautuivat pääosin mainostoimiston toimittamiin visuaalisiin mallikuviin. Muita määrittelyjä ei juurikaan tehty. Tehtävänhallintajärjestelmään ei siis kirjattu, mitä oli testattu tai mitä testauksessa oli havaittu. Näin ollen tehtävänhallintajärjestelmän seurannasta ei saatu muuta tulosta kuin, että sitä ei juurikaan hyödynnetty testauksessa tai sen dokumentaatiossa.

CI-palvelimelta kuitenkin näkyi, että versionhallintaan tehdyt päivitykset käännettiin ja asennettiin automaattisesti. CI-palvelin ilmoitti myös mahdollisista virheilmoituksista sähköpostitse ja tiimin käyttämälle keskustelukanavalle. CI-palvelimen havaitsemat ongelmat korjattiin tehokkaasti.

Projektitiimi ei pitänyt virallisia testaussessioita projektin aikana. Myöskään viikkotaiset tapaamiset eivät toteutuneet säännöllisesti. Projektitiimi kuitenkin kommunikoi epävirallisesti projektin aikana myös testaukseen liittyvistä asioista. Projektin järjestelmällinen seuranta ei kuitenkaan ollut mahdollista, sillä projektitiimi sopi ja kommunikoi asioista pääosin suullisesti ja epäformaalisti.

4.5. Loppuhaastattelut

Projektin seurannan lopuksi tekijöitä ja projektipäällikköä haastateltiin. Tämän avulla pyrittiin selvittämään oliko testauspolitiikan avulla muodostetusta testausstrategiasta hyötyä, kuinka testauspolitiikkaa voisi hyödyntää ja koettiinko politiikan sisältö sopivaksi.

Haastattelut olivat semistrukturoituja. Kysymykset oli jaoteltu neljään kategoriaan: projektin testaus, testauspolitiikan hyödyntäminen, testauspolitiikan sisältö ja haastateltavan tausta. Haastattelukysymykset kokonaisuudessaan ovat liitteessä 2.

Ensimmäisen kategorian kysymyksillä selvitettiin, kuinka projektissa oli tehty testausta ja oliko testausstrategiassa sovittuja testauskäytäntöjä noudatettu. Lisäksi projektin testausta verrattiin muihin projekteihin. Testauspolitiikan hyödyntämiseen liittyvissä kysymyksissä selvitettiin, miten testauspolitiikkaa voitaisiin hyödyntää ja oliko projektin testausstrategian luomisessa testauspolitiikasta hyötyä. Testauspolitiikan sisältöä käsittelevät kysymykset selvittivät, pitäisikö testauspolitiikan formaattia muuttaa ja kuvasiko testauspolitiikan sisältö haastateltavien mielestä yrityksen testauskäytäntöjä. Haastattelun lopuksi selvitettiin muutamalla kysymyksellä haastateltavan taustaa. Taustat on esitelty jo kohdassa 4.1.

Haastattelujen tulokset on selvitetty alakohdissa 4.5.1 - 4.5.3. Tulokset on jaoteltu kysymyskategorioiden mukaisesti.

4.5.1. Projektin testaus

Haastateltavat kertoivat projektin testauspanoksen jääneen hyvin vähäiseksi. Kun haastatteluissa käytiin läpi testausstrategiaan sovittuja käytäntöjä, vain harva niistä oli toteutunut. Haastateltavien mukaan testiautomaatio oli toteutunut suunnitellusti, mutta testauksen dokumentaatiota ei tehty ollenkaan. Projektissa ei myöskään oltu järjestelmällisesti testattu valmistuneita tehtäviä, vaan oli luotettu pohjalla olevan toteutuksen aikaisempaan testaukseen.

Haastateltavat kertoivat, etteivät pitäneet testaussessioita. Yksi koodikatselmointitilaisuus oli kuitenkin pidetty, ja tämän lisäksi projektin arkkitehti oli käynyt toteutuksia läpi. Projektin kehittäjät kertoivat, että varsinkin projektin alkuvaiheessa he tarkastivat toistensa tekemisiä ristiin. Projektipäällikkö kertoi myös käyneensä läpi

toteutuksia ja verranneensa niitä vaatimuksiin. Lisäksi projektipäällikkö oli tarkastanut uusien versioiden jälkeen, että järjestelmän perustoiminnallisuudet toimivat.

Kaikki haastateltavat olivat sitä mieltä, että projektin testauspanos oli todella pieni. Testauksessa oli keskitytty siihen, että sivusto näyttää mainostoimiston suunnitelmien mukaiselta, ja muilta osin oli luotettu pohjalla olevan toteutuksen testaukseen. Laatu arvioitiinkin yhtä hyväksi kuin muissa projekteissa, tai jopa paremmaksi perustuen projektin yksinkertaisuuteen.

Testausstrategiaan ei kirjattu mitään eri selaimilla ja päätelaitteilla testaamisesta. Kuitenkin varsinkin ulkoasun toteukseen keskittynyt kehittäjä oli tehnyt paljon testausta eri selaimilla ja päätelaitteilla. Muuten ei oltu testattu tavoilla, joita ei oltu kirjattu testausstrategiaan.

Asiakkaalle järjestettiin projektin seurannan aikana yksi koulutustilaisuus. Tässä tilaisuudessa asiakkaalle opetettiin sisällöntuotantojärjestelmänkäyttöä ja samalla neuvottiin testauksessa. Projektitiimi kertoi, että seurannan jälkeen on suunnitteilla lisää koulutus- ja testaussessioita yhdessä asiakkaan kanssa.

4.5.2. Testauspolitiikan hyödyntäminen

Kaikki haastateltavat kokivat, että testauspolitiikkaa voitaisiin hyödyntää kohdeyrityksessä. Kaikkien mielestä yritykselle muodostettu testauspolitiikka koettiin mahdolliseksi työkaluksi projektin testauksen suunnitteluun projektin alussa. Lisäksi osa haastatelluista toi esiin vaihtoehdon ottaa testauspolitiikka mukaan jo projektin myyntivaiheeseen. Haastateltavien mukaan testauspolitiikka helpottaisi myyntivaiheessa yrityksen testauskäytäntöjen kuvailemista. Haastateltavien kokemuksien mukaan myyntivaiheessa kuvataan aina uudestaan miten testausta tehdään, ja testauspolitiikka voisi ratkaista tämän ongelman.

Haastateltavat kokivat, että testauspolitiikka on hyvä muistilista projektien testauskäytöntöjen suunnitteluun. Heidän mukaansa testauspolitiikkaa olisi hyvä käydä myös asiakkaan kanssa kickoffissa eli projektin aloitustilaisuudessa läpi. Tällöin myös asiakas hyväksyisi testauskäyännöt ja sitoutuisi niihin.

Esimerkkiprojektin projektipäällikkö toimii kohdeyrityksessä myös projektipäällikkötiimin vetäjänä. Hänen mukaansa testauspolitiikkaa voitaisiin hyödyntää myös

uusien työntekijöiden perehdytyksessä. Etenkin ohjelmistokehittäjille ja projektipäälliköille olisi hänen mukaansa hyödyllistä selvittää kohdeyrityksen peruskäytännöt heti kohdeyrityksessä aloittaessa.

Vaikka haastateltavat kokivat, ettei testauspolitiikka ollut case-projektissa kovinkaan hyödyllinen, olisivat he valmiita hyödyntämään sitä muissa projekteissa. Projektipäällikkö aikoikin ottaa testauspolitiikan käyttöön toisen projektinsa aloitustilaisuuteen. Suurin osa haastateltavista olisi myös valmis suosittelemaan testauspolitiikkaa työkavereilleen.

4.5.3. Testauspolitiikan sisältö

Haastateltavien mielestä muistilista on hyvä muoto testauspolitiikalle. Yksi haastatelluista olisi kuitenkin kaivannut testauspolitiikkaan enemmän konkreettisia vaihtoehtoja. Muut sen sijaan olivat sitä mieltä, että testauspolitiikka on sopivan yleisellä tasolla. Haastateltavien mukaan yleinen taso on hyvä, sillä silloin se on helpoiten sovellettavissa kohdeyrityksen eri tyyppisiin projekteihin.

Kaiken kaikkiaan haastateltavat olivat sitä mieltä, että testauspolitiikan sisältö sopii hyvin kohdeyritykselle. Heidän mukaansa testauspolitiikka kuvaa sitä, miten testausta yleensä projekteissa tehdään. Lisäksi haastateltavat totesivat, ettei testauspolitiikassa ole liikaa sisältöä. Kuitenkin haastateltavat sanoivat, että kaikki tärkeimmät asiat ovat testauspolitiikassa huomioituna.

5. JOHTOPÄÄTÖKSET

Tässä luvussa esitellään työn tulokset kokonaisuudessaan. Lisäksi esitellään työn jatkosuunnitelmia ja mahdollisia jatkotutkimusideoita. Kohdassa 5.1 esitellään esimerkkiprojektin tulokset. Kohta 5.2 arvioi testauspolitiikan muodostamiseen käytettyä menetelmää. Lopuksi luvussa 5.3 esitellään työn jatkosuunnitelmat ja muutamia ideoita jatkotutkimukselle.

5.1. Testauspolitiikan hyödyntäminen case projektissa

Tässä työssä käytetty esimerkkiprojekti oli todella pieni ja pohjautui suurimmilta osin aiempaan toteutukseen. Tämän vuoksi tehty toteutustyö jäi todella vähäiseksi ja oli ulkoasupainotteista. Tästä johtuen testauspolitiikasta ei ollut juurikaan hyötyä esimerkkiprojektille.

Esimerkkiprojektin tiimi kuitenkin koki, että testauspolitiikka oli hyvä muistilista. Heidän mukaansa testauspolitiikassa oli testaukselle tärkeät asiat tiivisti ja yleispätevästi kerrottuna. Kuitenkin testauspolitiikan avulla muodostetun testausstrategian käytäntöjä noudatettiin heikosti. Projektitiimin mukaan tämä johtui siitä, että projektin testauspanokset jäivät ylipäänsä pieniksi.

Työssä käytetty esimerkkiprojekti oli liian pieni, jotta voitaisiin arvioida testausstrategian muodostamista testauspolitiikan avulla luotettavasti. Kaikkea suunniteltua seurantaa ei myöskään saatu toteutettua. Esimerkkiprojektin avulla saatiin kuitenkin kehitettyä menetelmää, jonka avulla testausstrategia voitaisiin muodostaa testauspolitiikan avulla. Tämän vuoksi seuraavien esimerkkiprojektien testausstrategian luominen sujuisi vähemmällä työllä. Lisäksi tämän työn pohjalta on tietoa mahdollisista mittareista arviointia varten.

5.2. Testauspolitiikan muodostaminen

Työssä muodostettiin testauspolitiikka perustuen yrityksen työntekijöiden haastatteluihin. Testauspolitiikka muodostettiin grounded theory -menetelmän mukaisesti haastatteluaineistoista tunnistettujen teorioiden pohjalta. Tämän menetelmän avulla saatiin muodostettua yritykselle testauspolitiikka. Testauspolitiikka vaikuttaa olevan riittävän kattava, mutta kuitenkin tarpeeksi yleispätevä sopiakseen alihankintaprojekteja tekevälle yritykselle.

Esimerkkiprojektissa työskennelleiden henkilöiden mukaan testauspolitiikan muoto ja sisältö sopivat hyvin yritykselle. Kattavampaa arviota varten tarvitaan kuitenkin arvioita laajemmin yrityksestä. Testauspolitiikka on kuitenkin kehityskelpoinen ja sitä voidaan alkaa koekäyttää yrityksessä kattavammin.

Testauspolitiikka on muodostettu täysin alhaalta ylöspäin. Tämän vuoksi testauspolitiikka perustuu täysin nykyisiin testauskäytäntöihin. Kuitenkin parempia tuloksia saavutettaisiin, jos haastateltaisiin myös yrityksen johdon edustajia. Näin varmistettaisiin, että testauspolitiikka tukee yrityksen tavoitteita ja strategiaa pitemällä tähtäimellä.

5.3. Jatkosuunnitelmat

Testauspolitiikan soveltaminen useammissa case-projekteissa. Testauspolitiikan sopivuutta testausstrategian luontiin olisi hyvä testata useammissa projekteissa. Aineistoa olisi hyvä kerätä erikokoisilta projekteilta eri toimialoilta, jotta voitaisiin muodostaa luotettava arvio muodostetun testauspolitiikan hyödystä testausstrategian luonnissa. Lisäksi testauskäytäntöjen toteutumista olisi järkevää verrata kohdeyrityksen projekteihin, joissa testauspolitiikka ei ole ollut käytössä. Tätä varten täytyy kehittää mittareita, joiden avulla käytäntöjen toteutumista voidaan vertailla systemaattisesti. Kohdeyrityksessä on lukuisia projekteja ylläpitovaiheessa. Useisiin näihin tehdään myös pienkehitystä ja korjauksia, mitkä tulee testata. Testauspolitiikan hyödyntämistä täytyy kokeilla myös ylläpitoprojekteihin, eikä pelkästään uusiin projekteihin. Lisäksi projektien seurannan tulisi olla pitkäkestoisempaa, jotta vaikutukset myös ylläpitovaiheeseen saataisiin selville.

Testauspolitiikan hyödyntäminen kohdeyrityksessä. Testauspolitiikan käyttöönot-

to kohdeyrityksen laajuisesti vaatii suunnitelmallisen lähestymistavan. Testauspolitiikkaa on suunniteltu hyödynnettävän myyntivaiheessa. Tarjouksiin vaaditaan usein kuvaus yrityksen testauskäytännöistä ja tähän testauspolitiikka sopii hyvin. Testauspolitiikka toimii testauksen muistilistana yksittäisten projektien testausta suunniteltaessa. Tätä varten testauspolitiikkaa tulee levittää projektipäälliköille, ohjelmistokehittäjille, käyttöliittymäsuunnittelijoille ja mahdollisille testaajille. Lisäksi tarvitaan selkeät ohjeet ja koulutusta siihen, kuinka testauspolitiikkaa voi hyödyntää testausstrategian muodostamiseen. Testauspolitiikka kuvaa yrityksen testaukseen liittyviä arvoja ja käytäntöjä. Nämä on hyvä selventää myös yrityksessä aloittaville uusille työntekijöille. Testauspolitiikan avulla uusien työntekijöiden perehdytys testauskäytäntöihin olisi mahdollisesti selkeämpää.

Asiakkaan havainnot testauspolitiikasta. Tämän työn puitteissa asiakkaan panokset testaukseen jätettiin huomioimatta. Kuitenkin kohdeyrityksen projektit tehdään lähes aina jollekkin asiakkaalle, joka osallistuu myös testaamiseen. Testauspolitiikkahaastatteluissa asiakkaan tekemä hyväksymistestaus nousi myös usein esiin. Jatkon kannalta olisi tärkeää selvittää myös asiakkaan kokemukset testauspolitiikan käytöstä. Tätä varten testauspolitiikka tulee esitellä myös asiakkaalle ja asiakas tulee sitouttaa osaksi testausta. Asiakkaalla on useimmiten viime kädessä paras toimialuetietämys. Asiakas myös hyväksyy projektin lopputuloksen ja dokumentaation. Tämänkin vuoksi testauskäytäntöjen luotettavaan arvioon tarvitaan myös asiakkaan panos.

Testauskäytäntöjen vertaileminen samankaltaisissa yrityksissä. Testauspolitiikan kehittämismenetelmää olisi mielenkiintoista soveltaa myös muissa kohdeyrityksen kaltaisissa yrityksissä. Näin saataisiin tietoa siitä, miten yrityksen kulttuuri ja arvot vaikuttavat testauskäytäntöihin. Lisäksi saataisiin mielenkiintoista tietoa esimerkiksi Tampereen tai koko Suomen parhaista testauskäytännöistä. Suomessa on jonkin verran samankaltaisia alihankintaprojekteja tekeviä yrityksiä, mutta samassa kokoluokassa olevien yritysten määrä ei kuitenkaan ole kovin suuri. Tämän vuoksi jo muutamaa yritystä tutkimalla otannasta saataisiin kattava.

Konsulttitalojen ja tuotetalojen testaamisen vertaileminen. Testauskäytäntöjen vertaileminen konsulttiyritysten ja tuotetalojen välillä toisi kiinnostavaa tietoa sii-

tä, kuinka testaaminen eroaa alihankintaprojekteissa ja yrityksen oman tuotteen kehityksessä. Tuotetaloissa testataan jatkuvasti samaa järjestelmää, minkä vuoksi testauskäytännöt saattavat olla vakiintuneempia. Sen sijaan konsulttiyrityksissä testataan erilaisia järjestelmiä ja testauskäytännöt saattavat vaihdella asiakkaasta ja projektista riippuen paljonkin. Tämän lisäksi voitaisiin vertailla eroja testauspolitiikan vastaanottamisessa erilaisissa yrityksissä. Eroja voi löytyä myös siinä, onko yrityksessä jo aikaisemmin ollut testauspolitiikka tai muuten vakiintuneet testauskäytännöt.

6. YHTEENVETO

Testauspolitiikalla tarkoitetaan organisaation laajuisia testauskäytöntöjä ja testauksen arvoja. Testausstrategia puolestaan on yhden projektin lähestymistapa testaukseen ja siinä kuvataan projektissa tehtävä testaus. Testauspolitiikan avulla voitaisiin yhteinäistää yrityksen testauskäytäntöjä ja sitä kautta huolehtia, että yrityksen toteuttamien järjestelmien laatu pysyy tasaisena.

Testauspolitiikan määrittämiselle ei ole selkeää menetelmää. Testauspolitiikka voidaan määrittää joko ylhäältä alaspäin tai alhaalta ylöspäin. Ylhäältä alaspäin -lähestymistavassa johto määrittää testauspolitiikan. Alhaalta ylöpäin -lähestymistavassa testauspolitiikka pohjautuu testausta tekevien työntekijöiden näkemyksiin.

Tämän työn tavoitteena oli määrittää Solita Oy:lle testauspolitiikka ja arvioida tätä esimerkkiprojektin testausstrategian luonnissa. Testauspolitiikka määritettiin alhaalta ylöspäin -lähestymistavalla perustuen yrityksen nykyisiin testauskäytäntöihin. Testauskäytännöt selvitettiin yrityksen työntekijöitä haastattelemalla, minkä jälkeen testauspolitiikkaa sovellettiin esimerkkiprojektin testausstrategian luomiseen.

Testauspolitiikasta pyrittiin saamaan kevyt ja käytännönläheinen. Testauspolitiikan täytyi myös olla tarpeeksi yleinen, sillä yritys tekee paljon erilaisia projekteja erilaisille asiakkaille. Tässä tavoitteessa onnistuttiin ja esimerkkiprojektin henkilökunta koki testauspolitiikan tarpeeksi yleispäteväksi.

Tässä työssä käytetty esimerkkiprojekti oli kooltaan pieni ja toteutustyö jäi vähäiseksi. Esimerkkiprojekti pohjautui voimakkaasti kohdeyrityksen aiemmin toteuttamaan projektiin, jota käytettiin työssä pohjana. Projektille onnistuttiin kuitenkin luomaan testausstrategia testauspolitiikan pohjalta. Tätä strategiaa ei kuitenkaan juurikaan projektissa noudatettu, johtuen ennakoituakin pienemmästä kehitys- ja testauspanoksesta. Näin ollen esimerkkiprojektista ei saatu riittävästi aineistoa tes-

tauspolitiikan luotettavaa arviointia varten.

Testauspolitiikan muodostaminen alihankintaprojekteja tekeville yrityksille ei ole helppoa, sillä testaustarpeet ja -käytännöt vaihtelevat paljon eri projekteissa. Testauspolitiikan luominen on kuitenkin mahdollista ja sen avulla voidaan helpottaa projektin testaustrategian luomista. Testauspolitiikkaa voidaan hyödyntää myös myyntiprosessissa ja uusien työntekijöiden perehdyttämisessä.

LÄHTEET

[1] C. Kaner and J. Bach. Context-driven-testing. `http://context-driven-testing.com/`. Accessed: 2016-11-21.

[2] Software and systems engineering software testing part 1:concepts and definitions. *ISO/IEC/IEEE 29119-1:2013(E)*, pages 1–64, September 2013.

[3] ISTQB International Software Testing Qualifications Board. *Standard Glossary of Terms used in Software Testing*, 3.1 edition.

[4] T. Kenny. From vision to reality through values. *Management Development Review*, 7(3):17–20, 1994.

[5] M. V. Mäntylä, J. Itkonen, and J. Iivonen. Who tested my software? testing as an organizationally cross-cutting activity. *Software Quality Journal*, 20(1):145–172, 2012.

[6] D. Martin, J. Rooksby, M. Rouncefield, and I. Sommerville. 'good' organisational reasons for 'bad' software testing: An ethnographic study of testing in a small software company. In *29th International Conference on Software Engineering (ICSE'07)*, pages 602–611, May 2007.

[7] J. Rooksby, M. Rouncefield, and I. Sommerville. Testing in the wild: The social and organisational dimensions of real world practice. *Computer Supported Cooperative Work (CSCW)*, 18(5):559, 2009.

[8] C. Andersson and P. Runeson. Verification and validation in industry - a qualitative survey on the state of practice. In *Proceedings International Symposium on Empirical Software Engineering*, pages 37–47, October 2002.

[9] A. Beer and R. Ramler. The role of experience in software testing practice. In *2008 34th Euromicro Conference Software Engineering and Advanced Applications*, pages 258–265, September 2008.

[10] C. Kaner, J. Bach, and B. Pettichord. *Lessons Learned in Software Testing : A Context-Driven Approach*. Wiley, Hoboken, 1 edition, 2001.

[11] J. Itkonen and K. Rautiainen. Exploratory testing: a multiple case study. In *Proceedings International Symposium on Empirical Software Engineering*, page 10, November 2005.

[12] O. Taipale, K. Karhu, and K. Smolander. Observing software testing practice from the viewpoint of organizations and knowledge management. In *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, pages 21–30, September 2007.

[13] I. Pinkster. Test policy: Gaining control on it quality and processes. In *Software Testing Verification and Validation Workshop, 2008. ICSTW '08. IEEE International Conference on*, pages 338–342, April 2008.

[14] Solita.fi. `http://www.solita.fi`. Accessed: 2017-04-15.

[15] R. Edwards and J. Holland. *What is Qualitative Interviewing?* Bloomsbury Publishing, 2013.

[16] A. Strauss and J. Corbin. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. SAGE Publications, 1990.

[17] Continuous integration. `https://www.thoughtworks.com/continuous-integration`. Accessed: 2017-05-07.

[18] Jira software. `https://www.atlassian.com/software/jira`. Accessed: 2017-05-07.

[19] A. Tarlinder. *Developer Testing, Building Quality into Software*. Addison-Wesley, 1 edition, 2016.

[20] Episerver. `http://www.episerver.com/`. Accessed: 2017-05-07.

[21] Jenkins. `https://jenkins.io/`. Accessed: 2017-05-07.

[22] What is a kanban board? `https://leankit.com/learn/kanban/kanban-board/`. Accessed: 2017-05-07.

[23] Google forms. `https://www.google.com/forms/about/`. Accessed: 2017-05-04.

A. LIITE 1: HAASTATTELUKYSYMYKSET

Testauspolitiikka:

- Mikä merkitys testauksella on Solitalle yrityksenä?
- Miten Solitan tavoitteet ohjaavat testausta?
- Miten testaus tukee projektin tavoitteita ja onnistumista?
- Miten projektin tavoitteet ohjaavat testausta?
- Miten testaus osa omaa työtä? (miten joutuu huomioimaan, hyödyntämään, suunnittelemaan)

Prosessit ja työkalut:

- Mitä kaikkea kuuluu testaukseen?
- Miten päätetään mitä testataan?
- Kenen vastuulla on laatu? Miksi?
- Minkälainen testausprosessi?
- Milloin testataan
- Miten testataan eri vaiheissa
- Minkälaisia suunnitelmia testaukselle on? Miten toteutunut? Miten poikennut?
- Miten testaus dokumentoidaan ja raportoidaan? Mitä tietoa tuotetaan
- Minkälaista testausta tehdään (yksikkötestit, järjestelmä, integraatio, tietoturva)? Millä menetelmillä?
- Mitkä on parhaita käytettyjä työkaluja?

- Minkälainen infrastruktuuri?
- Mistä on saatu apua?

Roolit:

- Kuka organisoi testauksen projektissa? Miten organisoi?
- Ketkä testaavat?
- Asiakkaan rooli?
- Mitä ominaisuuksia on hyvällä testaajalla?

Tausta:

- Mitä kuuluu työtehtäviin? Mitä tehnyt aiemmin
- Kauanko ollut Solitalla, muu työura? Paljonko projekteja?

B. LIITE 2: LOPPUHAASTATTELUKYSYMYKSET

Projektin testaus:

- Miten hyvin sovittuja testauskäytäntöjä noudatettiin? Mitä noudatettiin ja mitä ei? Miksi?
- Noudatettiinko käytäntöjä paremmin vai huonommin kuin muissa projekteissa sovittuja käytäntöjä? Millä tavoin? Miksi?
- Minkälainen projektin testauspanos oli muihin kokemiisi projekteihin verrattuna?
- Miten projektin laatu erosi musita projekteista?

Testauspolitiikan hyödyntäminen:

- Olisiko testauspolitiikka hyödynnettävissä Solitalla johonkin? Mihin ja miten?
- Kuinka paljon testauspolitiikasta oli hyötyä testausstrategian luonnissa? Miten? Miksi?
- Olisitko valmis hyödyntämään testauspolitiikka muissa projekteissasi ja suosittelemaan sitä myös työkavereillesi?

Testauspolitiikan sisältö:

- Mitä mieltä olet testauspolitiikan käskymuodosta? Olisiko joku muu muoto parempi?
- Onko testauspolitiikan sisältö Solitalle sopiva?

Tausta:

- Kuinka monta vuotta olet ollut Solitalla?
- Monesta projektista Solitalla sinulla on kokemusta? Entä kaiken kaikkiaan? Kuinka suuri osa ollut samankaltaisia?
- Mitä kuului työtehtäviisi projektin aikana?